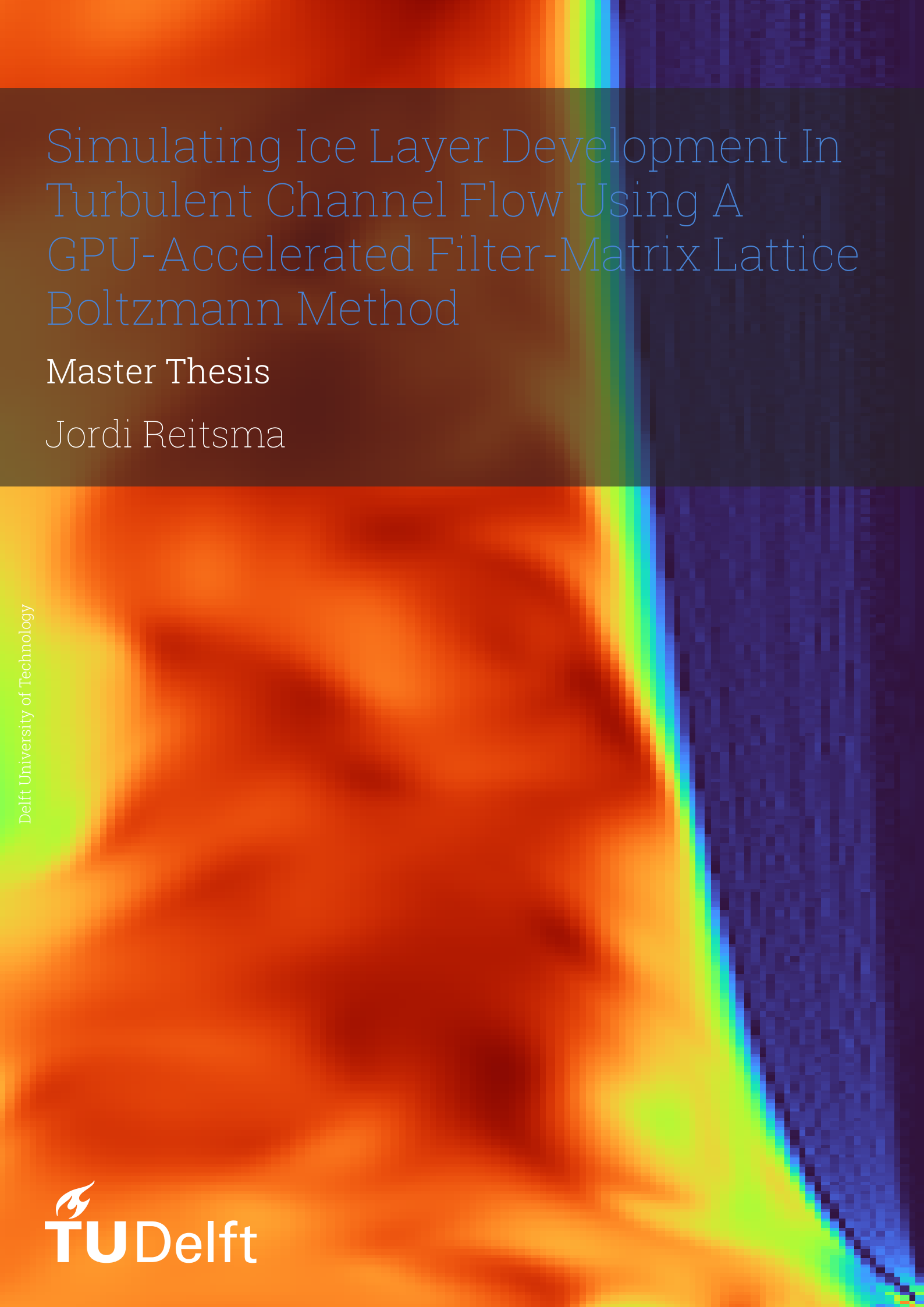
Simulating Ice Layer Development In Turbulent Channel Flow Using A GPU-Accelerated Filter-Matrix Lattice Boltzmann Method

Master Thesis

Jordi Reitsma



Delft University of Technology

TU Delft

Simulating Ice Layer Development In Turbulent Channel Flow Using A GPU-Accelerated Filter-Matrix Lattice Boltzmann Method

Master Thesis

JORDI REITSMA
4691245

A thesis submitted in partial fulfillment of
the requirements for the degree of

Master of Science in Applied Physics

To be defended on Thurdsday 17th of July 2025

Supervisors:	Dr. Ir. M. Rohde	
Examining commitee:	Dr. Ir. M. Rohde	Reactor Physics and Nuclear Materials, Applied Sciences, TU Delft
	Prof. Dr. Ir. J.L. Kloosterman	Reactor Physics and Nuclear Materials, Applied Sciences, TU Delft
	Dr. Ir. R.M. Hartkamp	Complex Fluid Processing, Mechanical Engineering, TU Delft

TU Delft

Reactor Physics and Nuclear Materials
Applied Physics
Delft University of Technology

Highlights

The most significant contributions of this thesis are:

1. *This work successfully implemented further GPU-acceleration by allocating the collision vector of the collision kernel into shared memory and using an optimized matrix mutliplication strategy.*
2. *This work created a DDF-FMLBM model that is able to simulate the ice growth behavior for a spatially developing ice layer.*
3. *This work compared freezing results from simulations with experimental data.*

ABSTRACT

The urgent need to combat climate change has focused research into low-emission energy technologies, with nuclear energy expected to play a pivotal role due to its reliability and minimal carbon footprint. Among the next-generation reactor designs, the Molten Salt Fast Reactor (MSFR) offers significant advantages in safety and efficiency. However, the high melting points of molten salts introduce a risk of freezing in the heat exchanger, which can severely impact reactor performance. To better understand possible freezing scenarios in the heat exchanger, this study develops a numerical model capable of simulating ice layer formation under turbulent flow conditions within channel flow.

The numerical model is based on a double-distribution function Filter-Matrix Lattice Boltzmann Method (DDF-FMLBM), enabling the accurate simulation of both turbulent flow and thermal phase change. The model includes enthalpy-based phase tracking and the immersed boundary method (IBM) imposes a no-slip condition within the ice. GPU acceleration is achieved via Julia's CUDA.jl, which enables the use of shared memory allocations which significantly improves the algorithm's efficiency.

The DDF-FMLBM implementation was validated against direct numerical simulation (DNS) literature studies, showing good agreement with benchmark turbulence statistics. Applying uniform grid refinement yielded only marginal accuracy improvements, whereas increasing the domain length enhanced large-scale eddy resolution in the streamwise direction, particularly improving the prediction of RMS velocity fluctuations (u'_{rms}). A strong recycling method was used to generate realistic turbulent inflow profiles in combination with a zero-gradient Neumann boundary as the outflow. It was found that the Neumann condition introduced non-physical effects near the outlet. GPU performance was evaluated in terms of MLUPS, and substantial speed-ups were achieved over previous implementations of DDF LBM models by making use of shared memory allocation for the FMLBM solution vector and optimized matrix multiplication strategy.

The developed DDF-FMLBM model was shown to accurately resolve key aspects of turbulent heat transport and phase change behavior in water. This validated model was used to simulate spatially developing ice layers under varying cold-wall temperatures, capturing ice thickness growth over a physical time of 10 minutes. Most cases did not reach steady-state freezing within this period. Comparison with a single experimental reference curve revealed similar freezing trends, though the simulation consistently over predicted the freezing rate. Due to mismatched experimental conditions, this discrepancy could not be quantitatively assessed. Residual velocities observed within the ice layer suggest that the realistic turbulent inflow condition using the strong recycling method introduces non-physical artifacts, as the immersed boundary method (IBM) performs correctly in periodic setups.

Based on the findings, several recommendations are proposed to enhance future modeling efforts. To enable validation at relevant conditions, the model should be extended to higher Reynolds numbers using Large Eddy Simulation (LES) to reduce computational cost. It is also advised to replace the zero-gradient Neumann outlet with more physical boundary conditions, such as convective outflow. Additionally, extending the model to support higher Prandtl numbers and adiabatic wall conditions would allow better comparison with experimental studies.

NOMENCLATURE

ABBREVIATIONS

Abbreviation	Definition
MSFR	Molten Salt Fast Reactor
LBM	Lattice Boltzmann Method
FMLBM	Filter-matrix lattice Boltzmann method
DNS	Direct Numerical Simulation
DDF	Double-distribution function
GPU	Graphical Processing Unit
LBGK	Lattice Bathnagar-Gross-Krook
MRT	Multi-relaxation time
IBM	Immersed boundary method
ETT	Eddy turnover time
MLUPS	Million lattice updates per second
RMS	Root mean square

CONTENTS

1 INTRODUCTION

In December 2015 the United Nations Framework Convention on Climate Change (UNFCCC) organized a conference on climate change attended by representatives of 196 nation states [1]. Here, a global agreement has been formulated to tackle the global temperature rise due to climate change. The main goal of the Paris Agreement is that the increase in global average temperature should be kept well below 2.0 °C above pre-industrial levels, while pursuing efforts to limit the temperature increase to 1.5 °C above pre-industrial levels.
In 2025 there are signs that global temperature is rising faster than predicted leading to 2023 and 2024 being the hottest years on record [2] [3]. The Aeronautics and Space Administration (NASA) reported that the global temperature rise in 2024 even surpassed the 1.5 °C threshold agreed upon in the Paris Agreements. Evidently, drastic measure need to be taken in order to uphold the Paris Agreements.

Carbon emissions represent the most significant factor contributing to climate change. Specifically, the current global energy consumption roughly emits 75% of the total fossil fuel emissions worldwide[4]. Therefore, the energy sector has received enormous research attention as innovation in this sector plays a pivotal role in tackling climate change. According to the International Energy Outlook 2023 formulated by the U.S. Energy Information Administration (EIA), the CO_2-emissions are expected to rise until 2050 due to growing global population and growing incomes. The most straightforward way to offset these rising emissions is by focusing innovation on creating fossil fuel free energy technologies.

Currently, nuclear energy created in a nuclear reactors produces electricity with the lowest amount of greenhouse emissions. To achieve a world where carbon emissions should be net zero, nuclear reactors are therefore crucial. Besides being a very clean energy source, it is also a very reliable way to produce energy. In a world with growing uncertainty, nuclear energy can help countries to move to become self sufficient in their own energy needs.

In a report published in 2023, the International Atomic Energy Agency (IAEA) estimated the share of nuclear energy to the total electricity mix could further grow from 9.8% to 14% [5]. Despite being a sustainable and reliable way of producing electricity, governments and institutions around the world have been reluctant to commission the construction of new reactors. Fear of major accidents and accumulation of nuclear waste are the main culprits that negatively impact public opinion on nuclear energy. To unlock the potential that nuclear energy holds to fight climate change, the Generation IV Forum (GIF) was established as an international cooperation on nuclear research between 13 countries as well as the European Union. The main goal of GIF is to develop the research necessary to commercialize six new prospective reactor types by 2030 [6]. Testing safety risks and performance are the main obstacles that have to be overcome to make the new reactors commercially attractive. One promising reactor type is the Molten Salt Reactor (MSR), which relies on molten salts serving as the reactor fuel, coolant and/or moderator.
This introductory chapter will first discuss the main principles of the MSR and discuss its advantages and disadvantages compared to conventional PWR's in section 1.1. Furthermore, section 1.2 presents the relevant studies from literature used as basis in this thesis. Section1.3 presents the research questions that this work aims to answer. To conclude the introduction, an outline for the rest of this thesis is given Section 1.4.

1.1 MOLTEN SALT FAST REACTOR

The nuclear reactor type that is most widely used today is the pressurized water reactor (PWR), which constitutes roughly 70% of the current nuclear fleet and generates 79% of all nuclear produced electric energy [7]. In a PWR, the fissile material is usually 3% enriched UO_2 rods , both the moderator and coolant are water. Two water loops are generally used within the nuclear reactor system. The primary water loop circulates through the reactor core and indirectly transfers its heat to the secondary water loop creating steam. Consequently, this steam drives a turbine that converts the steam heat into electrical energy. [8]

Even though PWRs have been the staple of nuclear energy generation for decades, the reactor type has inherent limitations concerning safety, fuel utilization and nuclear waste production. The moderator and coolant needs to be in the liquid phase to ensure proper reactor operation. As a result, water needs to be pressurized increasing the explosion risk during meltdown. Another consequence is that the reactor core can only operate effectively for temperatures up to the fairly low boiling point of 340 °C at operating pressures. This in turns leads to a limit on thermal efficiency of 34%. Other drawbacks include low fuel utilization, long-lived nuclear waste and the need for active safety mechanisms.

The molten salt reactor (MSR) type reactors aim to improve on the major drawbacks of the PWR, while still being capable of producing a viable amount of (electrical) energy. The defining feature, is the use of molten salt as a coolant and/or moderator and in many designs, also as the medium in which fissile material is dissolved. Of particular interest to this study is the molten salt fast reactor (MSFR) concept envisioned by the SAMOSAFER (Severe Accident Modeling and Safety Assessment for Fluid-fuel Energy Reactor) project, since Delft University of Technology is a member of the projects consortium [9]. A simplified design of the MSFR can be seen in figure 1.1.

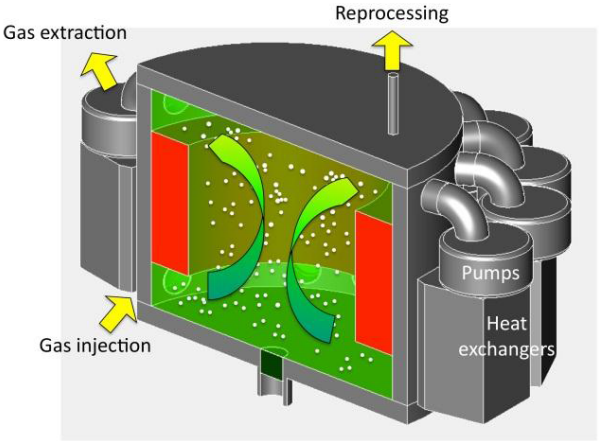


Figure 1.1: Schematic of the working principle of the MSFR envisioned by the SAMOSAFER project. The reactor core is represented by the green area. The location of heat exchanger and pumps is also specified. [9]

The reactor core (in green) is contained within a cylindrical vessel of height and diameter of $2.25m$ and is filled with nuclear fuel dissolved in a salt. Within the vessel, a breeding blanket (in red) is present, which envelops the reactor core. The blanket contains thorium, which enables the breeding of more fissile material[9, 10]. Fuel salt from the core is circulated in a downward direction through the heat exchangers via pumps. Consequently, the flow of the fuel salt within the reactor core is upwards. No moderator is present, meaning that fission reactions are driven by fast neutrons, enabling the reactor to operate in the range of a breeder or burner reactor.

Using fuel salts comes with multiple advantages over PWR's. First of all, the thermal expansion of the fuel salt, due fission reactions, has a negative impact on the fission rate serving as a natural feedback system when the reactor gets overheated. Furthermore, a higher thermal efficiency can be achieved which is attributed to the high boiling point (at $740C^{\circ}$) of the molten salt. Other advantages include the possibility of fuel adjustment during operation, better resource utilization, fuel homogeneity and passive safety features like a freeze plug [7].

1.1.1 Risk of freezing

A major complication in the current state of MSFR research is the fact that molten salts are prone to freezing, because of their high melting points. This is specifically a concern in the heat exchanger, where heat from the reactor core is extracted from the molten salt mixture. Solidification within the heat exchanger can have catastrophic effects on the operation of a MSFR [11]. These effects are twofold: (1) the material of the heat exchanger may be damaged due to volumetric expansions of the molten salt mixture and (2) the ice layer can partially or completely block the flow from the reactor, reducing the system's potential for efficient heat transfer.

To better understand scenarios where freezing takes place, it is essential to model the effects of the phase change behavior of these molten salts. Current MSFR designs estimate that the core flow can operate at Reynolds numbers between the orders of magnitude 10^3 and 10^5, thus placing the flow in a turbulent state [12]. It is therefore important to investigate the effects of turbulence on the development of an ice layer within the heat exchanger. The present numerical study focuses on the phase change behavior under turbulent flow conditions.

1.2 PREVIOUS RESEARCH

This thesis aims to create a numerical model that is able to model ice layer development in a heat exchanger. The specific numerical technique of interest is the lattice Boltzmann method (LBM) which is able to solve the governing equations for the flow- and thermal physics. One of the attractive features of the LBM is that it is suitable for implementation on a Graphical Processing Unit (GPU), which can dramatically increase computational speed. An increase in computational speed is particularly advantageous for simulating turbulent flows, which often require stringent requirements on spatial resolution and domain size to capture flow physics [13]. The Lattice Boltzmann method originates from the lattice gas automata (LGA) method and is based on kinetic theories [14]. LBM modeling revolves around finding a suitable way to model the collision operator. The earliest successful collision model is the Bhatnagar-Gross-Krook (LBGK) model, which relaxes particle distributions to equilibrium using a single-relaxation-time(SRT) [15, 16]. A major drawback of this model is that it suffers from numerical instability, especially at low viscosity [17], which is a significant concern when modeling turbulence. An improvement over the SRT-model is the multiple-relaxation-time (MRT) approach, which assigns distinct relaxation times to different moments of the distribution function [18]. This increases the numerical stability and accuracy of the LB model [19]. However, finding the right relaxation parameters in MRT is difficult, which are needed to achieve stable and accurate results[20]. Somers et al. [21] introduced a novel lattice Boltzmann scheme in which the nonlinear collision operator is modeled with the assistance of filter matrices. This Filter-Matrix Lattice Boltzmann (FMLBM) approach retains the advantages of multiple relaxation times, while eliminating the need to specify individual relaxation parameters. In both studies from literature [19, 22, 23]and studies conducted within the research group involved in this thesis [24–26], the FMLBM is proven to be an accurate LB model for simulating both turbulent thermal and flow dynamics.

A successful implementation of thermal effects within the FMLBM framework was demonstrated by Zhuo et al. [19]. A double-distribution function (DDF) approach was used, that simulates the flow field and temperature field using separate distribution functions. This DDF strategy offers improved numerical stability and allows for consistent thermal boundary treatment, compared to other thermal LBM formulations. To implement a transition from liquid to ice in the thermal model, Huang et al. [27] proposed replacing the temperature based thermal distribution function with a distribution function that evolves total enthalpy. This formulations allows modeling the effects of the extraction or absorption of latent heat associated with phase transitions. To keep track of the moving phase interface, the immersed boundary method (IBM) was used developed by Noble et al. [28]. The implementation of phase change effects within the FMLBM framework was shown to be successful by Bus [26] and Spek [25]. Specifically, the latter study focused on simulating turbulent freezing using a D3Q19-D3Q19 DDF-FMLBM approach. It was demonstrated that instabilities in the thermal field attributed to large enthalpy values, could be overcome by implementing suitable enthalpy transformation rules.

Modeling realistic turbulent freezing requires a suitable turbulent inflow condition. Chung et al. [29] showed that using a temporal strong recycling method reliably reproduces all relevant turbulent characteristics of the target flow, making it suitable for generating realistic inflow conditions. Within the associated research group, Collenteur [30] conducted an experimental investigation on convective freezing in turbulent and laminar channel flow using PIV measurements to identify ice layer growth. The findings from this study serve as a reference for validating the freezing model in the current work.

1.3 RESEARCH QUESTIONS

The focus of this thesis is put on developing a computationally efficient FMLBM model that can simulate realistic freezing within a channel flow, that mimics the geometry of the heat exchanger of the MSFR. To this end, the following research questions are formulated for this thesis.

1. **Methodology:** *How can transient salt freezing under turbulent flow conditions in cooled 3D channel be modeled using a GPU-accelerated direct numerical simulations (DNS) in the filter-matrix lattice Boltzmann (FMLB) framework in order to best resemble experimental results?*
    a) *Which boundary-, inlet- and outlet conditions should be implemented to achieve stable and accurate simulations?*
    b) *) What is the appropriate grid configuration to realistically capture all relevant turbulent scales in DNS?*
    c) *To what extent can computational efficiency be improved while maintaining accurate results?*
2. **Ice layer development:** *How and to what extend is the thickness and profile of a solidified ice layer in a turbulent flow through a cooled 3D channel affected by imposed thermal boundary conditions and cold wall temperature?*

1.4 THESIS OUTLINE

The current chapter has given the motivation for developing MSFR's, the advantages and disadvantages of these novel reactor types, provided literature relevant to this specific study and set the goal for the rest of this work. Chapter 2 will provide the theoretical background that is needed to understand the relevant physics and concepts involved in developing the DDF FMLBM model. Furthermore, the specific implementation steps needed to arrive at an accurate and efficient thermal freezing model are outlined in Chapter 3. Subsequently, the usefulness of simulating flow field specifics of the model is tested in Chapter 4. In Chapter 5 heat transfer and phase change functionalities are validated and simulations on a developing ice layer are performed. The final chapter of this thesis summarizes the conclusions found from the results and gives recommendations on interesting future research paths.

2 THEORETICAL BACKGROUND

This chapter will provide the necessary background information that can be used to understand the concepts described in later sections. The first section 2.1 of this chapter discusses the governing equations associated with the field of fluid dynamics. After that section 2.2.2 presents relevant theory needed to understand phase change phenomena. Furthermore, kinetic theory related to the Lattice Boltzmann method is provided in section 2.3.1. Section 2.4 provides the relevant information on turbulence in fluid dynamics. Lastly, theory on parallel computing is presented in section 2.5

2.1 FLUID DYNAMICS

Fluid dynamics is the domain that is concerned with retrieving a correct description of the behavior of fluids. Correctly simulating the hydrodynamical behavior of a fluid in a channel begins with finding the governing equations of fluids. These equations can be found by considering the mass- and momentum conservation for a fluid element.

2.1.1 Mass equation

To be able to derive the governing equations, the continuum hypothesis for fluid dynamics is being employed. This hypothesis states that fluids can be treated as continuous matter rather than a constituency of microscopic particles. As result, continuous macroscopic variables can be defined at every point in time and space. [31]

Consider a three-dimensional macroscopic fluid element that moves through space with a three-dimensional velocity $\mathbf{u}$, which is defined by $\mathbf{u} = (u, v, z)$. Here u, v, and z correspond to the velocity components in the x-, y- and z-directions respectively. When there is no source or sink within the element, the mass balance or the continuity equation will read [31]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{2.1}$$

Here ρ is the density of the fluid element in kg/m^3. The first term in the equation is the local change of density, while the second term corresponds to the transport of mass. When considering an incompressible flow, the continuity equation reduces to:

$$\nabla \cdot (\rho \mathbf{u}) = 0 \tag{2.2}$$

The incompressibility assumption dictates that the density of the fluid is a constant. This assumption is valid when the fluid velocity is sufficiently small compared to the speed of sound, $\frac{u}{c_s}$

2.1.2 Momentum equation

Another important characteristic of fluid is how it evolves into space. To derive an expression for the momentum behavior, once again consider a control volume. Newton's second law states that the change in linear momentum is equal to the forces acting on the fluid element. On a fluid element specifically, only surface forces and volume forces apply. Using momentum conservation and the result from equation 2.2, the incompressible momentum conservation equation can be written as

$$\mu \nabla^2 \mathbf{u} - \nabla p + \rho \mathbf{f} = \rho \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}\right) \tag{2.3}$$

Here μ is called the dynamic viscosity in $Pa \cdot s$, p is the pressure in Pa and $\mathbf{f}$ is an acceleration in m/s^2 . In the current study it is also assumed that the fluid is Newtonian. This means that viscosity has a constant value [32]. The combination of equations 2.1 an 2.3 is also called the Newtonian Navier-Stokes equations for incompressible flows.

The left hand side of the momentum equation represents the effect of forces on the fluid. The term $\mu\nabla\mathbf{u}$ represents diffusion of momentum and is also called the viscous term. Viscosity tends to resist sharp gradients and acts as an inhibitor to momentum diffusion within the fluid. The second term in equation 2.3 represents a pressure force in the form of a gradient. Besides that, fluid motion can also influenced by body forces, which is embodied by $\rho\mathbf{f}$.

On the other hand, the right-hand side of equation 2.3 represents the inertial terms. What makes the Navier-Stokes equation notoriously hard to solve is the non-linear advection term $(\mathbf{u}\cdot\nabla)\mathbf{u}$. The main characteristic of this term is that small velocity fluctuations are amplified rapidly.

Solutions to the Navier-Stokes equations can be characterized by its Reynolds number

$$Re = \frac{UL}{\nu} \tag{2.4}$$

which is a non-dimensional number which represents the ratio between the inertial forces and the viscous forces. Here U is the characteristic velocity of the flow, L the characteristic length scale and ν the kinematic viscosity, which is defined as $\nu = \frac{\mu}{\rho}$. At a certain Reynolds number a flow will start to transition from laminar to a turbulent situation. For a typical pipe flow, this transition will begin around $Re = 2300$, where fully turbulent flow is observed for $Re > 2700$.[31, 33]. The nature of the transition of laminar to turbulence will be discussed in section 2.4.

2.2 THERMODYNAMICS

The subject of freezing fluids involves the process of heat transfer between different fluid phases. To arrive at a useful formulation of phase change, it is necessary to first derive the governing equation of heat transfer based on the concept of energy conservation. Subsequently, phase change behavior can then be quantified.

2.2.1 Thermal energy equation

Similarly to the derivation of the Navier-Stokes equation, a finite volume element is considered alongside an energy balance equation. In this energy balance the effects of viscous dissipation are neglected,the fluid is still considered incompressible and the specific heat for a particular phase is assumed to be constant. The thermal energy equation then looks like:

$$\rho C_p \frac{\partial T}{\partial t} + \rho C_p \nabla\cdot(\mathbf{u}T) = \nabla\cdot(\lambda\nabla T) + q \tag{2.5}$$

Here C_p, λ, T are the specific heat, thermal conductivity, fluid temperature and heat sink/source respectively [27]. As can be seen Eq. 2.5 takes a similar shape as the momentum equation Eq. 2.3. On the left hand-side the first term represents the transient term again, while the second term expresses the change in thermal energy as a result of convection. The first term on the right-hand side is the diffusion of thermal heat. The final term represents a heat source or sink that corresponds to the energy that is being absorbed or released due to phase change.

Equation 2.5 can also be cast into an enthalpy form, when considering that $dh = c_p dT$ [34]

$$\rho\frac{\partial h}{\partial t} + \rho\mathbf{u}\cdot\nabla h = \nabla\cdot(\alpha\nabla h) + q \tag{2.6}$$

A useful non-dimensional number that relates the momentum diffusivity ν to thermal diffusivity α is the Prandtl number Pr:

$$Pr = \frac{\nu}{\alpha} \tag{2.7}$$

The Prandtl number is a material specific number. When Pr » 1 then momentum diffusivity dominates over thermal diffusivity. This means that in order for heat to be transported efficiently, convection is needed. On the other hand when Pr « 1, heat is transported primarily through diffusion. [32]

2.2.2 Phase change

In the framework of phase change it is more convenient to track the total enthalpy H within the system. Fortunately, the sensible enthalpy can be directly related to the total enthalpy via: [34]

$$H^{\phi} = h^{\phi} + f_l^{\phi} L \tag{2.8}$$

The first term represents sensible enthalpy and the second term is the latent heat contribution. Essentially, latent heat reflects the amount of energy that is needed to change the molecular structure during phase change. Three distinct phases can be identified during the solidification process, namely the liquid, solid, and mushy phase. These phases are indicated in the formula using $\phi = \{l, s, m\}$. Furthermore, f_l is the liquid fraction, which can take values $0 \leq f_l \leq 1$. When the substance is fully liquid, its liquid fraction will be 1. Conversely, a solid phase will have a value of 0. Within the mushy zone the solid and liquid phase coexist. To account for the effects of both phases, a liquid fraction between 0 % 1 is allowed.

During the phase change the incompressibility assumption fails physically, since the solidification process involves volumetric expansion. To circumvent this problem, the source term q can be modeled to account for the phase change effects without the need to correct for density variations. For a pure material like water, q can be described as the change in total enthalpy [27]

$$q = -\frac{\partial(\rho \Delta H)}{\partial t} = -\frac{\partial(\rho L f_l)}{\partial t}. \tag{2.9}$$

For a pure material the latent heat is a constant [32]. As a result the advection term can be approximated by$\mathbf{u} \cdot \nabla H^{\phi} \approx \mathbf{u} \cdot \nabla h^{\phi}$. Using this and putting equation 2.8 into the enthalpy equation 2.6, the expression for total enthalpy of phase change is expressed as

$$\frac{\partial H^{\phi}}{\partial t} + \mathbf{u} \cdot \nabla h^{\phi} = \nabla \cdot (\alpha^{\phi} \nabla h^{\phi}). \tag{2.10}$$

The material being studied is considered to be eutectic. This means that the transition from one phase to the other happens at constant temperature. What differs in distinct phases, is the value for the specific heat. This results in a less straightforward way to calculate solution variables. Values for the liquid fraction and temperature can be retrieved via

$$f_L = \begin{cases} 0 & H < H_s \\ \frac{H - H_s}{H_l - H_s} & H_s \leq H \leq H_l \\ 1 & H > H_l \end{cases}$$

$$T = \begin{cases} H / C_{p,s} & H < H_s \\ T_s + \frac{H - H_s}{H_l - H_s}(T_l - T_s) & H_s \leq H \leq H_l \\ T_l + (H - H_l)/C_{p,l} & H > H_l, \end{cases}$$

where $C_{p,s}$ and $C_{p,l}$ are the specific heats for the liquid and solid phase. The solidus temperature T_s can be described as the highest temperature at which a solid phase can exist. Equivalently, the liquids temperature is the lowest temperature at which the liquid phase can exist. The same rationale applies to the total solidus enthalpy H_s and the total liquidus enthalpy H_l. For eutectic fluids $T_s = T_l$, but $H_s \neq H_l$. In the mushy layer the temperature will be constant, but the total enthalpy changes as a result of the addition or removal of latent heat.

Since the mushy zone is a mixture between the solid and the liquid phase, the specific heat of the fluid should also be adjusted. The specific heat in the mushy layer is defined to be $C_{p,m} = \frac{2C_{p,l}C_{p,s}}{C_{p,l}+C_{p,s}}$. Using this result enables to formulate the following expression for the sensible enthalpy

$$h = \begin{cases} C_{p,s} T, & T < T_s \\ h_s + C_{p,m} T, & T_s \leq T \leq T_l \\ h_l + C_{p,l} T, & T > T_l \end{cases} \tag{2.11}$$

where h_s and h_l are the solidus- and liquidus sensible enthalpy respectively.

2.3 KINETIC THEORY

2.3.1 Boltzmann equation

The governing principles of the lattice Boltzmann method is rooted in kinetic theory, which gives a description on a mesocopic scale of dillute gasses. The most important parameter in kinetic theory is the particle distribution function $f(\mathbf{x}, \boldsymbol{\zeta}, t)$, which represents a distribution of particles at position $\mathbf{x}$, velocity $\boldsymbol{\zeta}$ and time t. In this mesoscopic approach, collection of particles is tracked rather than individual particles. Macroscopic variables can be retrieved from the particle distribution function, by taking its moments[35]:

$$\rho(\mathbf{x}, t) = \int f(\mathbf{x}, \boldsymbol{\zeta}, t) d^3\zeta \tag{2.12}$$

$$\rho(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) = \int \boldsymbol{\xi} f(\mathbf{x}, \boldsymbol{\xi}, t)\, \mathrm{d}^3\xi. \tag{2.13}$$

The evolution equation of the particle distribution function is retrieved by considering its derivative with respect to time and expanding its partial derivative terms:

$$\frac{\partial f}{\partial t} + \xi_\beta \frac{\partial f}{\partial x_\beta} + \frac{F_\beta}{\rho} \frac{\partial f}{\partial \xi_\beta} = \Omega(f). \tag{2.14}$$

The above equation is referred to as the continuous Boltzmann equation. Here F_β is a force term. The collision operator on the right-hand side represents a source term, which describes the redistribution of f towards local equilibrium:

$$\Omega(f) = -\frac{1}{\tau}(f - f^{eq}) \tag{2.15}$$

Here f_{eq} represents the equilibrium distribution function. This particular collision operator was invented by [15] and it represents the relaxation towards equilibrium governed by the relaxation time τ.

2.4 TURBULENCE

2.4.1 What is turbulence?

Solutions to the Navier-Stokes equation can either give a laminar, flow profile or a turbulent profile. A laminar profile is characterized by an orderly and smooth arrangement of fluid layers as shown in the top of figure 2.1 . In such a flow the viscous term $\mu\nabla^2\mathbf{u}$ dominates. The viscosity is large enough to suppress any effect of perturbations in the flow field, stabilizing the flow such that the fluid layers stays orderly.

Turbulent flow on the other hand looks chaotic and disordered as can be seen in the bottom picture. The orderly fluid layers seem to have broken down and started to mix, creating circular structures called eddies. In three-dimensional space, the eddies possess a vortex shape. In the case of turbulent flows, the non-linear inertial term $(\mathbf{u}\cdot\nabla)\mathbf{u}$ has a bigger effect than the viscous term on overall flow. As a result perturbations no longer get damped out, but rather get enhanced. In turn this creates eddies. Even though turbulent flow looks random, it is still fully deterministic [31]

.

Within a turbulent flow, a whole range of different eddy sizes and shapes are present. In turn, all these different eddies interact and influence each other to create a variety of turbulent structures. The largest scales are associated with the *macro-structure*, which is characterized by large swirling motions within the flow field. By means of these large eddies, quantities like temperature are effectively mixed. Therefore, turbulence can be an effective mechanism for heat transfer.

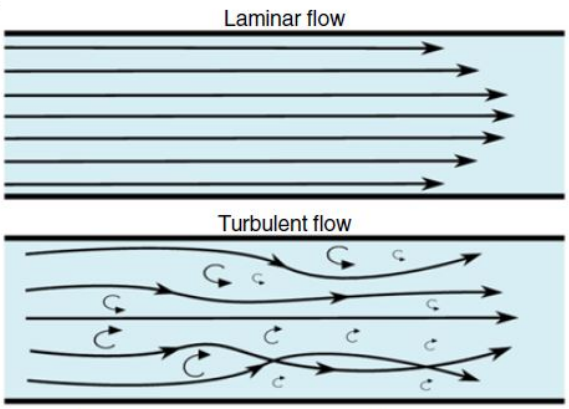


Fig. 2. Difference between laminar and turbulent flow

Figure 2.1: Comparison between laminar flow and turbulent flow [36]

Another important characteristic of turbulent flow is the existence of the energy cascade, which is depicted in figure 2.2. Energy is injected into the flow at large scales by external forces, generating large eddies that eventually become unstable. These unstable eddies break down into smaller vortices in a process that continues down to the smallest eddy scales, known as the *microstructure*. At these small length scales, viscous effects become dominant due to the increasing velocity gradients. As a result, the energy originating from the large eddies is dissipated by the fluid's viscosity. The microstructure is directly associated with the kinetic energy dissipation rate ϵ. Important parameters to quantify the microstructure are the Kolmogorov scales [31]

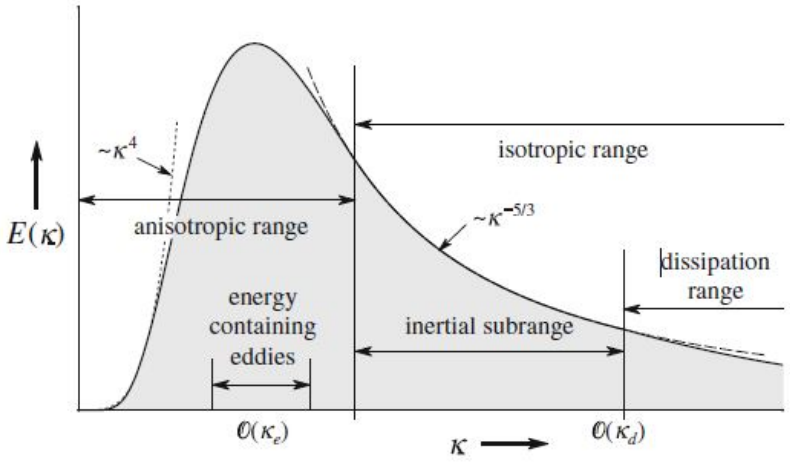


Figure 2.2: Schematic depiction of the energy casade within a turbulent flow [37].

$$\eta = \left(\frac{\nu^3}{\epsilon}\right)^{1/4}, \quad \tau = \left(\frac{\nu}{\epsilon}\right)^{1/2}, \quad \upsilon = (\nu\epsilon)^{1/4}. \tag{2.16}$$

Here η is called the Kolmogorov length scale and is associated with the smallest eddy size. Furthermore, τ and υ are the characteristic timescale and velocity scale associated with the Kolmogorov length scale. The range which the eddy scales can take, is related to Reynolds number via

$$\frac{\mathscr{L}}{\eta} \sim Re^{3/4}. \tag{2.17}$$

Here $\mathscr{L}$ is the characteristic length scale associated with the macro-structure. When the Reynolds number grows, ratio between the biggest scales and the smallest scales increase.

2.4.2 Turbulence Statistics

As discussed, turbulent flows are characterized by a highly chaotic and fluctuating flow field. Small deviations in boundary conditions or initial conditions can yield different instantaneous flow profiles. Retrieving useful information , thus requires a statistical analysis for both the solutions to the Navier-Stokes equations as well as the ones for the heat equation.

In turbulent statistical theory it is common to define an instantaneous quantity as the sum of an average and a fluctuation

$$u_i = \overline{u_i} + u_i', \tag{2.18}$$

Here the overhead bar denotes the mean of a quantity, while the apostrophe represents a fluctuation. These velocity This decomposition is called the Reynold decomposition and forms the basis for calculating turbulence statistics.

The mean velocity $\overline{\mathbf{u}}$ is defined as the ensemble average

$$\overline{\mathbf{u}} = \lim_{N\to\infty} \frac{1}{N} \sum_{\alpha=1}^{N} \mathbf{u}^{(\alpha)}, \tag{2.19}$$

where the same simulation is performed N times. Index α represents one realization of the experiment. Turbulent channel flow is considered to be statistically stationary and homogeneous, because the solutions are both invariant to translations in both space and time [31]. As a consequence, the spatial- and temporal averages are both equal to the ensemble average. Thus, the mean quantity in this study is averaged over both space and time,

$$\overline{u}(y) = \frac{1}{N_x N_z N_T} \sum_{i=1}^{N_x} \sum_{j=1}^{N_z} \sum_{n=1}^{N_T} u(x_i, y, z_j, t_n), \tag{2.20}$$

$$\overline{T}(y) = \frac{1}{N_x N_z N_T} \sum_{i=1}^{N_x} \sum_{j=1}^{N_z} \sum_{n=1}^{N_T} T(x_i, y, z_j, t_n), \tag{2.21}$$

Here the resulting mean quantity is a function of y and thus no averaging takes place over the wall-normal direction.

A way to quantify the intensity of turbulence within a flow is given by the root mean square (RMS) velocity fluctuation defined as u_{rms}. In statistical terms tis RMS values are defined by

$$u_{rms} = \sqrt{\overline{u'u'}}, \quad T_{rms} = \sqrt{\overline{T'T'}}. \tag{2.22}$$

Here T_{rms} is the temperature equivalent to the RMS velocity fluctuations and it quantifies thermal mixing within a fluid. u_{rms} is directly related to the turbulent kinetic energy via

$$k = \frac{1}{2}(\overline{u_i' u_i'}) \tag{2.23}$$

An important feature of turbulent flow is the effective transport of momentum and heat transport due to turbulent eddies. This transport primarily occurs due to shearing effects , where velocity gradients generate fluctuations that enhance mixing. Statistically, this mixing effect can be captured by

$$\tau_{ij} = -\rho \overline{u_i' u_j'}, \quad q_i = \rho c_p \overline{u_i' T'}, \tag{2.24}$$

where τ_{ij} is referred to as the Reynolds stress tensor and q_i as the turbulent heat flux.

The presented quantities are also referred to as the low-order statistics. Higher-order statistics involve higher order moments of the velocity and temperature values. However, heat transfer mechanisms resulting from turbulence are confined solely to the presented lower-order statistics. Therefore it is not necessary to consider these higher-orders statistics.

2.4.3 Channel flow

Turbulent flows within a channel like the one shown in figure2.3 are characterized by the shearing effect of the both walls. The no-slip condition imposed on the walls create a region where viscous and turbulent effects interact, leading to distinct near-wall behavior. This type of flow is also referred to as a turbulent boundary layer flow. The channel half-height is denoted by the symbol H. Fully developed turbulent profiles within a channel are stationary in time and horizontally homogeneous, which implies zero time gradients and zero spatial gradients in the x- and z-directions [31].

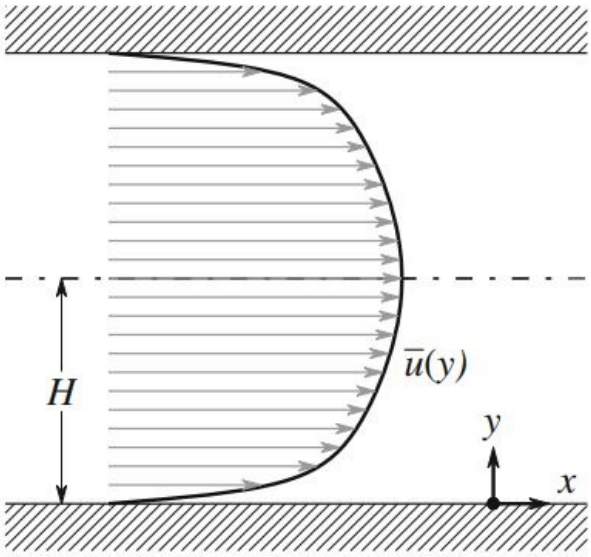


Figure 2.3: Two-dimensional representation of the geometry in turbulent channel flow. Here the parallel plates are seperated by a distance 2H. $\overline{u}$ corresponds the characteristic mean velocity of this flow type [31].

The characteristic Reynolds number for channel flow is defined by

$$Re_m = \frac{2HU_m}{\nu}. \tag{2.25}$$

Here U_m is the velocity mean taken over the wall-nomal distance of the channel.
Using Prandtl's mixing length closure hypothesis, an analytical expression can be derived for the mean velocity profile. This hypothesis depends on defining the so-called mixing length $\mathscr{L}$, that can be used to find an analytical expression for the shear stress. The mixing length can be interpreted as the characteristic length scale of an eddy. Because of the presence of the walls, the eddy size is a function of the distance to the wall. Based on this observation, three distinct region can be identified with each their unique solution profiles and mixing lengths.

These regions are the *core region*, *wall region* and the *viscous sublayer*. Characteristic of the core region is that the characteristic eddy size scales with the channel height and that the contribution of "small" eddies is negligible. The Reynold stress tensor dominates over the viscous stress, meaning the flow is considered fully turbulent here.

Getting closer to the wall from the core region means that the characteristic eddy sizes shrink and that the relative contribution of the viscous stress tensor grows. The core regions transitions into the *logarithmic wall layer*. As the name suggest the solution to the velocity profile here is logarithmic. Furthermore, the turbulent stress is constant, while the viscous stress is still negligble.

In the region closest to the wall eddies are small enough for the viscous stress to become dominant over Reynold stress. The flow in this region is effectively laminar, but turbulent effects are induced because of the presence of the other two regions. Since viscous forces are relatively strong here, this part of the channel has been named the *viscous sublayer*.

Analytical solutions for each region can be obtained by matching solutions of different zones at appropriate locations. The mean velocity profile near the wall is defined as

$$u^+ = \begin{cases} y^+ & 0 < y^+ < 5, \\ \frac{1}{k}\left[\ln\left(y^+\right) + \Pi\right] & y^+ > 30. \end{cases} \tag{2.26}$$

Here the constant $k \approx 0.4$ is the van Karman's constant. $\Pi = 2.0$ is an offset constant following from the matching of solution of the viscous sublayer and logarithmic layer at $y^+ = 11$. Here y^+ is the wall-distance in wall-units, which will be explained shortly. The region $y^+ < 5$ corresponds to the viscous sublayer and from $y^+ > 30$ this layer transitioned into the logarithmic wall layer. Between $5 \geq y^+ \leq 30$ a buffer layer is present where both the viscous stress and turbulent stress are important[31]. Furthermore the logarithmic wall region ends at around $y^+ = 100$, from where the core region begins

Wall units
A convenient way to quantify the near-wall behavior in channel flows is to cast the relevant variables into non-dimensional wall-units. An universal scaling system enables comparison between different channel flow setups exhibiting the same turbulent behavior.

Relevant wall units can be formulated for different variables involved. The general procedure is to find suitable characteristic near-wall value for the quantity and subsequently scale the original value with it. In the near-wall region, the viscous stress dominates and determines the characteristic wall velocity:

$$\tau_w = \rho\nu\left(\frac{d\overline{u}}{dy}\right)_{y=0}, \quad u_\tau = \sqrt{\frac{\tau_w}{\rho}}. \tag{2.27}$$

Here τ_w represent the wall shear stress. Wall length is expressed by the viscous length scale

$$\delta_\nu = \frac{\nu}{u_\tau} \tag{2.28}$$

which is the fundamental scale over which momentum diffusion happens near the wall. Plugging the viscous length scale and the friction velocity into the formula for the Reynolds number yields:

$$Re_\tau \equiv \frac{u_\tau H}{\nu} \tag{2.29}$$

Here Re_τ is called the friction Reynolds number and describes how near-wall inertial effects behaves compared to the viscous effects.

Wall units for position, velocity and time are then given by

$$y^+ \equiv \frac{y}{\delta_\nu}, \quad u^+ \equiv \frac{u}{u_\tau}, \quad t^+ = \frac{u_\tau}{H}t, \tag{2.30}$$

where the plus sign signifies that the variable is in wall units.

A suitable wall unit for temperature can also be defined based on the friction temperature T_f, which in turns leads to wall temperature T^+

$$T_f \equiv -\frac{\alpha}{u_\tau}\left(\frac{d\overline{T}}{dy}\right)_{y=0}, \quad T^+ = \frac{T}{T_f}. \tag{2.31}$$

2.4.4 Turbulence Simulation Techniques

A direct numerical simulation (DNS) of turbulent flow aims to resolve all the turbulent scales that are present within the flow problem, both in space and time. In other words, DNS solves the full Navier-Stokes equations without any turbulence modeling. As a result, DNS retrieves a unique and highly-accurate solution that fully represents the unsteady dynamics of turbulence.[38] To achieve a high-fidelity DNS, the computational domain should be large enough to accommodate the movement of the largest eddies in all directions. Furthermore, the information of smallest spatial scales corresponding to the Kolmogorov length should be captured by the computational grid. Analogous requirements should be met considering to accurately capture the timescales [13]. Specifics for the simulations performed in this thesis are discussed in Section 3.5
As mentioned the main advantage of DNS is that it can represent the solutions of the NSE with very high accuracy. Besides that no modifications of the NSE or modeling is needed to capture relevant physical effects. Its biggest advantage accuracy wise also leads to its drawbacks on the computational level. Equation 2.17 shows that the Kolmogorov size decreases for flows with higher Reynolds number using the same channel configuration. Consequently, the grid size needs to be refined leading to more computational nodes in the domain. Simulating very high Reynolds number flows is therefore a big challenge for DNS. Besides the computational costs, memory costs also significantly increase when more grid points are used.

2.5 PARALLEL PROGRAMMING ON A GRAPHICAL PROCESSING UNITS

Simulating turbulent channel flow at high Reynolds numbers is computationally demanding, because resolving the full-range of turbulent scales requires increasingly finer grids. Reducing computational costs is thus critical in turbulent models. Traditionally these calculations run on the central processing unit (CPU), which updates grid nodes sequentially. However, LBM operations are inherently local, since updating a computational cell requires only adjacent grid nodes. A graphics processing unit (GPU) can handle these type of local operations far more efficiently than a CPU, paving the way for faster simulation times [35]. The aim of this section is to introduce key theoretical concepts that are needed to understand the GPU-implementation of the current FMLBM model.

2.5.1 GPU hardware architecture

A schematic depiction of the most important hardware components of an NVIDIA GPU is shown in Figure 2.4. A GPU contains multiple streaming multiprocessors (SMs) and on device global memory. The SMs are individual processing units, which are responsible for managing resources across CUDA cores and executing parallel tasks. Additionally, SMs contain a block of shared memory that allows operations on the same processing unit to share resources. A warp scheduler is also present in the SM, which manages the coordination of the parallel instructions. Lastly, the SM contains thousands of CUDA cores that are responsible for performing calculations. Each CUDA core has a small amount of on-core register memory.

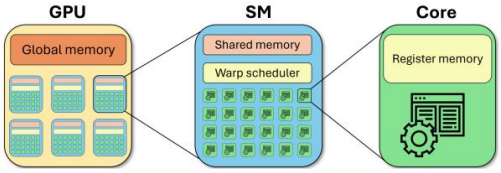


Figure 2.4: Schematic depiction of the standard architecture of an NVIDIA GPU. The picture shows the hierarcal structure of the most important hardware components of the GPU. The GPU contains multiple SMs and global device memory (RAM). Each SM contains its own shared memory cache, a warp scheduler and thousands of CUDA cores. These CUDA cores serve as the main arithmetic units in the GPU and have access to on-core register memory.

2.5.2 Parallel programming and CUDA

In this research, the Compute Unified Device Architecture (CUDA) programming language is used to leverage the parallel computing power of NVIDIA GPUs. In CUDA, kernels serve as the GPU equivalent to functions in CPU-programming and are designed to perform the same operation across many grid points. When a kernel is launched, threads are created. These threads are software abstractions that carry the kernel's instruction for a specific grid point. The information of the thread is executed by the CUDA cores. Due to the GPU's hardware structure, threads are bundled together in groups of 32 threads, called a warp. A collection of warps is again grouped into a block. The SMs are capable of scheduling and assigning blocks, which can contain millions of threads, to be executed by the CUDA cores. To launch CUDA kernels, the amount of blocks and threads per block needs to be specified. Achieving high computation performance relies on choosing an optimum values for the kernel configurations.

The CPU and the GPU are distinct hardware components within a computer and in programming are treated as such. Functions and input data need to be defined so as to be compatible for use on the GPU. To achieve this, the standard workflow for parallel programming with CUDA is as follows:

1. Allocate GPU memory
2. Transfer input memory from CPU to GPU
3. Run parallel simulation using GPU
4. Transfer data back from GPU to CPU

Parallel CUDA-programming introduces several programming difficulties that are not present in CPU programming. One such difference is that an individual CUDA core is not as powerful as an individual CPU core. Kernels should therefore not contain expensive computations. The GPU relies on a high throughput of tasks in order to outperform a CPU. As result of this, high level functionality is not directly supported within GPU kernels. For example, a matrix multiplication is not supported in the CUDA framework, requiring for loops to calculate this operation. Consequently, code within a kernel is fairly low-level and limited in functionality as compared to CPU code. As a result, parallel efficiency depends heavily on the ability to design kernels that optimize the effective use of the GPU's hardware.

Another common phenomena encountered in parallel-programming are race conditions. When multiple threads access and modify the same memory location,the the final outcome of the kernel depends on the order of execution. Now, each time the kernel is launched a different thread can be the last one to access the memory location, leading to unpredictable simulation results. To achieve accurate simulation results, race conditions should therefore be avoided.

2.5.3 GPU memory hierarchy

Understanding the memory hierarchy inherent to the GPU hardware is crucial in optimizing the performance of GPU workloads. An overview is provided of the most relevant memory types used in parallel programming:

1. **Global memory**: Main GPU storage pool which is accessible by all threads in a GPU. Access to global memory is slow compared to other memory levels.
2. **Constant memory** Small amount of memory that is read only. Located in global memory and relatively slow.
3. **Local memory** Memory that is specific to each thread. Local memory stores temporary variables or spill from overfull registers. Local memory accesses is slower than accessing registers.
4. **Shared memory** High speed memory shared among threads in blocks. Primarily used for inter-thread communication.
5. **Register memory** Smallest amount of memory and is assigned to single CUDA core. Registers are the fastest memory spaces within the hierarchy

3 NUMERICAL METHODS

The Filter Matrix Lattice Boltzmann method will to model the governing equations of fluid dynamics and thermodynamics. First, the Lattice Boltzmann method and the Filter-Matrix approach for fluid flow simulations will be presented in section 3.1 . The model will then be extended in section 3.2 to include phase change functionality using a double-distribution function (DDF) approach. Next, boundary conditions of interest are discussed in section 3.3. Furthermore, the parallel programming implementation on the GPU will be presented in section 3.4. Finally, the specifics of the Direct Numerical Simulation approach is presented in section 3.5.

3.1 FILTER–MATRIX LATTICE BOLTZMANN METHOD

3.1.1 Lattice Boltzmann Method

The discrete-velocity distribution function $f_i(\mathbf{x}, t)$ is the main quantity of interest when considering LBM schemes. It represents the density distribution function that was discussed in Section 2.3.1, but then projected on a discretized lattice with uniform grid spacing Δx . The discrete function f_i also evolves over time with a timestep of Δt .The most convenient choice for the lattice step size and lattice timestep are $\Delta x = 1\,ls$ and $\Delta t = 1\,lt$, where ls stands for lattice spacing and lt lattice time.

Different velocity sets are allowed when discretizing the velocity space. Figure 3.1 shows two different schemes, which are denoted by DdQq. Here d is the number of spatial dimensions, while q represents the number of velocities considered. Discrete velocities are depicted as arrows pointing to neighboring lattice points, with each velocity having a distinct direction. For simplicity the D2Q9 is shown, alongside the more complicated D3Q19 velocity set which will be used in this research.

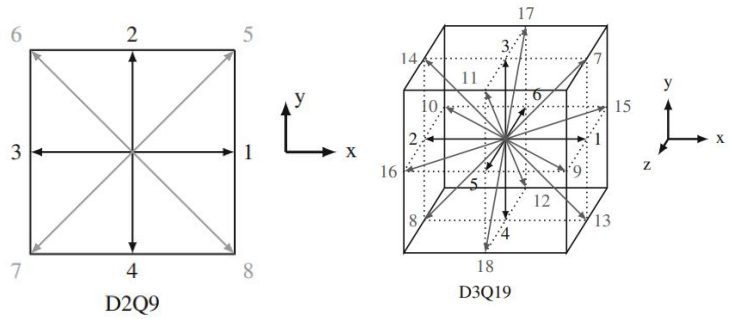


Figure 3.1: Visualisation of the D2Q9 (left) and the D3Q19 (right) velocity sets [35]

Discretizing the Boltzmann evolution equation from Eq. 2.14 will lead to the following expression:

$$f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) + \Delta t \Omega_i(\mathbf{x}, t). \tag{3.1}$$

This equation is called the lattice Boltzmann equation and represents the effects of two fundamental processes within the LBM. The first one is the collision process, which is characterized by the collision operator Ω. Fundamentally, the collision operator models how fluid particles interact and relax towards an equilibrium solution. After collision, the distributions will move to neighboring sites based on their discrete velocities. This process is called the propagation step and is incorporated into Eq. 3.1 by the left hand-side. The post-collision distribution function is streamed to the location $\mathbf{x} + \mathbf{c}_i \Delta t$ and to next timestep $t + \Delta t$. A schematic depiction of the general procedure of an LBM algorithm is shown in figure 3.2.

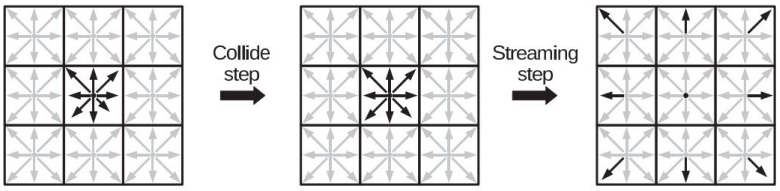


Figure 3.2: Schematic representation of the two fundamental processes of an LBM algorithm. Each node contains all distinct distribution functions. In the collision step these distributions get redistributed locally, to be propagated to neighboring nodes in the propagation step [39]

Besides streaming and collision, imposing boundary conditions on the system is also a crucial step in the LBM algorithm. Section 3.3 will delve deeper into specific boundary conditions that have been used. Relevant simulation variables can be obtained from the simulation by considering the different moments of the particle distribution function

$$\rho(\mathbf{x}, t) = \sum_i f_i(\mathbf{x}, t) \tag{3.2}$$

$$\rho\mathbf{u}(\mathbf{x}, t) = \sum_i \mathbf{c}_i f_i(\mathbf{x}, t) \tag{3.3}$$

where the zeroth-order moment corresponds to the retrieval of the fluid density ρ and the first-order moment retrieves the momentum. Higher order moment also have a physical meaning, but are not relevant in the current study.

Right now it is not trivial that equation that 3.1 can represent the solutions to the Navier-Stokes equations given a suitable collision operator and distribution function. A Chapmans-Enskog analysis [40] is needed to relate both equations to each other. This derivation shows that the Navier-Stokes Equations can be retrieved from the lattice Boltzmann equation when the distribution function solution has the following form [41]:

$$f_i = f_i^{eq} - \rho\omega_i \left(\frac{(\mathbf{c}_i \cdot \nabla)(\mathbf{c}_i \cdot \mathbf{u})}{c_s^4} - \left(1 + \frac{2}{D} - B\right) \frac{\nabla \cdot \mathbf{u}}{c_s^2} \right) \tag{3.4}$$

Here D represents the spatial dimension of the simulation, B is the ratio between bulk viscosity and kinematic molecular viscosity $B = \frac{\zeta}{\nu}$ and c_s is the speed of sound. Furthermore, f_{eq} is the equilibrium distribution function and to retrieve the Navier-Stokes equation is defined as follows:

$$f_i^{eq} = \omega_i \rho \left(1 + \frac{\mathbf{c}_i \cdot \mathbf{u}}{c_s^2} + \frac{(\mathbf{c}_i \cdot \mathbf{u})^2}{2c_s^4} - \frac{\mathbf{u} \cdot \mathbf{u}}{2c_s^2} \right). \tag{3.5}$$

3.1.2 Filter-Matrix Lattice Boltzmann Method

As discussed in the introduction, the Filter-Matrix Lattice Boltzmann Method (FMLBM) offers excellent stability for simulating high Reynolds number flows. Essentially, the need for a relaxation parameter to model the collision operator is removed by making use of filter matrices and corresponding solution vectors. Following the derivation described in [42], the LBM-equation is first recast into a staggered form

$$f_i\left(x + c_i \frac{\Delta t}{2}, t + \frac{\Delta t}{2}\right) - f_i\left(x - c_i \frac{\Delta t}{2}, t - \frac{\Delta t}{2}\right) = \Delta t \Omega_i. \tag{3.6}$$

This staggering ensures that both the numerical accuracy in time- and space is second-order without needing to apply extra numerical treatments. The staggered formulation can be used alongside equations 3.4 and 3.5 to find an expression for the non-linear collision operator

$$\Omega_i = \frac{\rho\omega_i}{c_s^2} \left[(\mathbf{c}_i \cdot \nabla)(\mathbf{c}_i \cdot \mathbf{u}) - c_s^2 \nabla \cdot \mathbf{u} \right] + \frac{\omega_i}{c_s^2} \mathbf{c}_i \cdot \mathbf{f}. \tag{3.7}$$

where $\mathbf{f}$ is the body force vector. Employing a first-order Taylor expansion around staggered distribution function $f_i\left(x \pm c_i \frac{\Delta t}{2}, t \pm \frac{\Delta t}{2}\right)$ and using equations 3.1 gives the following relation

$$f_i\left(\vec{x} \pm \frac{\vec{c}_i \Delta t}{2}, t \pm \frac{\Delta t}{2}\right) = f_i(\vec{x}, t) \pm \frac{1}{2}\Delta t \Omega_i \tag{3.8}$$

It turns out that the right hand side of the equation can be formulated as a matrix multiplication like

$$f_i\left(x \pm c_i \frac{\Delta t}{2}, t \pm \frac{\Delta t}{2}\right) = \sum_k w_i E_{ki} \alpha_k^{\pm}(x, t), \tag{3.9}$$

where E_{ki} is a reversible filter matrix and $\alpha_k^{\pm}(x,t)$ represents the solution vector. The minus sign represent the pre-collision solution vector and the plus sign denotes the post-collision solution vector. Since E_{ki} is a reversible matrix, equation 3.9 can also be rewritten in the following form:

$$\alpha_k^{\pm}(x, t) = \sum_k E_{ki} f_i\left(x \pm c_i \frac{\Delta t}{2}, t \pm \frac{\Delta t}{2}\right). \tag{3.10}$$

Here $E_{ki} = (\omega_i E_{ik})^{-1}$ represent the inverse of the filter-matrix.
The appropriate filter matrix and solution vector for simulating the Navier-Stokes Equation is uniquely determined by the velocity set used.

3.1.3 D3Q19 velocity Scheme

In this study the D3Q19 velocity scheme as depicted on the right-hand side of figure 3.1 will be used. This velocity scheme has already shown its ability to successfully retrieve reliable turbulence statistics in several studies [42] [25]. Another velocity set that bears great potential of simulating turbulence flow is the D3Q27 scheme [43], however it has not been considered in this work since it aso requires more computational costs.
The appropriate filter matrix and solution vector for simulating the Navier-Stokes Equation using a D3Q19 velocity scheme are given by

$$\begin{aligned} E_{ki} = \Big[& 1, c_{ix}, c_{iy}, c_{iz}, 3c_{ix}^2 - 1, 3c_{iy}^2 - 1, 3c_{iz}^2 - 1, \\ & 3c_{iy}c_{iz}, 3c_{ix}c_{iz}, 3c_{ix}c_{iy}, 3c_{ix}(c_{iy}^2 - c_{iz}^2), 3c_{iy}(c_{iz}^2 - c_{ix}^2), 3c_{iz}(c_{ix}^2 - c_{iy}^2), \\ & c_{ix}(|c_i|^2 - 2), c_{iy}(|c_i|^2 - 2), c_{iz}(|c_i|^2 - 2), \\ & 3(|c_i|^2 - \frac{3}{2}), 3|c_i|^2(|c_i|^2 - 2) + 1 \Big]^{\mathrm{T}}. \end{aligned} \tag{3.11}$$

$$\alpha_k^{\pm} = \begin{bmatrix} \rho \\ \rho u_x \pm \frac{\delta_t F_x}{2} \\ \rho u_y \pm \frac{\delta_t F_y}{2} \\ \rho u_z \pm \frac{\delta_t F_z}{2} \\ 3\rho u_x^2 + \rho(-6\nu + \delta_t)\partial_x u_x + (2 - 3B)\rho\vec{\nabla} \cdot \vec{u} \\ 3\rho u_y^2 + \rho(-6\nu + \delta_t)\partial_y u_y + (2 - 3B)\rho\vec{\nabla} \cdot \vec{u} \\ 3\rho u_z^2 + \rho(-6\nu + \delta_t)\partial_z u_z + (2 - 3B)\rho\vec{\nabla} \cdot \vec{u} \\ 3\rho u_y u_z + \rho(-3\nu + 0.5\delta_t)(\partial_y u_z + \partial_z u_y) \\ 3\rho u_x u_z + \rho(-3\nu + 0.5\delta_t)(\partial_x u_z + \partial_z u_x) \\ 3\rho u_x u_y + \rho(-3\nu + 0.5\delta_t)(\partial_x u_y + \partial_y u_x) \\ -0.8, \quad k = 11, \ldots, 16 \\ -0.95, \quad k = 17, 18, 19 \end{bmatrix}, \quad k = 1, 2, \ldots, 19. \tag{3.12}$$

where $c_s = 1/\sqrt{3}$ is the speed of sound, ω_i are the weight factors for each velocity direction corresponding to $\omega_1 = 1/3$, $\omega_{2-7} = 1/18$ and $\omega_{7-19} = 1/36$. Furthermore, B is a parameter that relates the bulk viscosity and the local kinematic viscosity. The values -0.8 and -0.95 are parameters used to increase stability in turbulent simulations.

3.1.4 Conversion Parameters

In LB simulations it is common to define the simulation step size and timestep to be $\Delta x = 1$ and $\Delta t = 1$ respectively . Therefore unit conversion is needed to map the physical input variables to the discretized lattice. A good knowledge of LB unit conversion is needed, especially to maintain stability and accuracy within the LB simulation. An important restriction on the stability of the simulation is that the density distribution function should not become too big. Concretely, the maximum LB velocity U_{LB} generally should not exceed the value 0.1 during the simulations [35].

Expressing physical units in lattice units or LB-units can be done by formulate the right conversion parameter C. A physical quantity like length l can be denoted in different units, like meters or feet. The key here is to find the right conversion parameter between the two length scales. Similarly, conversion from physical units to lattice-units comes down to find the correct conversion parameter. Converting length, time and density to LB-units comes down to

$$\Delta x_{LB} = \frac{\Delta x}{C_l}, \quad \Delta t_{LB} = \frac{\Delta t}{C_t}, \quad \rho_{LB} = \frac{\rho}{C_\rho}, \quad T_{LB} = \frac{T}{C_T}. \tag{3.13}$$

Here the subscript LB denotes a variable is represented in lattice units. In addition to that C_l, C_t, C_ρ, and C_T are the conversion parameters for length, time, density, and temperature respectively. The conversion factors for any other relevant quantity α can be found by matching its unit via

$$C_\alpha = C_l^a C_t^b C_\rho^c C_T^d. \tag{3.14}$$

The letters a, b, c , d can take any integer value. In this study $\Delta x_{LB} = 1$, $\Delta t_{LB} = 1$ and $\rho_{LB} = 1.0$, which means that the conversion parameters for these variables correspond to the physical value.Since there are no stability restrictions on the thermal distribution function g, C_T is taken to be unity. Enthalpy values however still need conversion, because its dimensions are related to velocity

$$h_{LB} = \frac{C_l^2}{C_t^2} h \tag{3.15}$$

3.2 FILTER-MATRIX THERMAL LATTICE BOLTZMANN METHOD

3.2.1 Double-distribution function

Besides being a reliable method to solve the Navier-Stokes equation, Huang et al. [27] also showed that the Lattice Boltzmann method is suitable for solving the heat equation. Simultaneously simulating the flow field and the thermal field can be done through multiple methods by employing a double-distribution function (DDF) method [27]. This approach relies on simulating an additional, separate distribution function—specifically for heat transfer—denoted by the thermal distribution function g_i which governs the evolution of the temperature field. Similarly to equation 3.1 , the thermal LBM is captured by

$$g_i(\mathbf{x}+\mathbf{c}_i\Delta t, t+\Delta t) = g_i(\mathbf{x}, t)+\Omega_i(\mathbf{x}, t). \tag{3.16}$$

which will be referred to as the thermal lattice Boltzmann equation. Similar to equation 3.1, this equation captures the collision and the propagation inherent to the LBM algorithm in as single statement. The thermal equilibrium distribution function reads

$$g_i^{\mathrm{eq}} = \omega_i\, T\left[1+\frac{\mathbf{c}_i\cdot\mathbf{u}}{c_s^2}+\frac{(\mathbf{c}_i\cdot\mathbf{u})^2}{2\,c_s^4}-\frac{\mathbf{u}^2}{2\,c_s^2}\right]. \tag{3.17}$$

Now the zeroth order moment gives the macroscopic temperature

$$T(x,t) = \sum_{i=0}^{N} g_i(x,t). \tag{3.18}$$

From the definition of the equilibrium function it is clear that the temperature field depends on the velocity field. Within the DDF-approach it is therefore important to first update the density distribution function f_i and subsequently update the thermal distribution function g_i using the flow field information.

3.2.2 Enthalpy-distribution

As established in section 2.2.2 phase change phenomena are associated with a change in latent heat and can best be described by total enthalpy. Consequently, the current description of the thermal LBE is not suitable for solidification purposes. Following [27], an LB algorithm can be devised that simulates total enthalpy by making an adjustment to the equilibrium distribution function

$$g_i^{eq} = \begin{cases} Lf_l + w_0 h(1-\frac{u^2}{2c_s^2}) & i=0 \\ w_i h[1+\frac{\mathbf{c}_i\cdot\mathbf{u}}{c_s^2}] & i\neq 0 \end{cases}$$

Here it can be observed that the equilibrium distribution function has been changed such that $i=0$ includes the latent heat term Lf_l. The benefit of splitting the total enthalpy into such a way is avoiding to use iterative schemes to calculate the liquid fraction [27]. Such iterative schemes are computationally expensive. During the collision step the latent heat term is subtracted. Adding the non-equilibrium part to this equation yields the total thermal distribution function

$$g_i = \begin{cases} Lf_l + w_0 h(1-\frac{u^2}{2c_s^2}) & i=0 \\ w_i[h+h\frac{\mathbf{c}_i\cdot\mathbf{u}}{c_s^2}-\alpha\frac{\mathbf{c}_i\cdot\nabla h}{c_s^2}] & i\neq 0 \end{cases}$$

Because of the adjustments, the total enthalpy H can now be retrieved from the zeroth order moment of the thermal distribution function

$$H(\mathbf{x}, t) = \sum_i g_i(\mathbf{x}, t) \tag{3.19}$$

To find suitable expression for the thermal collision vectors, the density distribution function f_i in the derivation shown in section 3.1.2 is replaced by g_i. This leads to the following formulations

$$g_i\left(\mathbf{x}\pm\frac{\mathbf{c}_i\Delta t}{2}, t\pm\frac{\Delta t}{2}\right) = \sum_k w_i E_{ik}\beta_k^{\pm}(\mathbf{x}), \tag{3.20}$$

$$\beta_k^{\pm}(\mathbf{x}, t) = \sum_k E_{ki}\, g_i\left(\mathbf{x}\pm\frac{\mathbf{c}_i\Delta t}{2}, t\pm\frac{\Delta t}{2}\right). \tag{3.21}$$

for the staggered thermal distribution function and the thermal collision vectors $\beta^{\pm}$

Several heat transfer studies have been performed using LBM with a D3Q7 velocity scheme. However, it was found in [25] and also in this study that for turbulent heat transfer simulations the D3Q7 is not sufficient in suppressing higher-order errors. Therefore an D3Q19 thermal scheme was also adopted in the current study. The matrix E_{ki} takes the same values as equation 3.11 . However, the collision vector $\beta^{\pm}$ in this case looks like

$$\beta_k^{\pm} = \begin{bmatrix} h \\ hu + \frac{-8\alpha\pm\Delta t}{8}\partial_x h \\ hv + \frac{-8\alpha\pm\Delta t}{8}\partial_y h \\ hw + \frac{-8\alpha\pm\Delta t}{8}\partial_z h \\ 0, \quad k = 5,\ldots,19 \end{bmatrix}.$$

where the first term represents sensible enthalpy and the terms 3 till 5 correspond to heat advection and heat diffusion in space. Furthermore, the non-physical terms corresponding to element 5 till 19 are all set to zero. The benefit of using a D3Q19 scheme instead of D3Q7 is the effect of these higher order elements, which ensure the Galilean Invariance is preserved. They surpress any higher-order oscillations leading to higher stability within the thermal field [35].

3.2.3 Enthalpy scaling

Several studies that use an enthalpy-based DDF-LBM approach to simulate thermal fields, observed non-physical enthalpy fluctuations when using the physical temperature as an input parameter. [25], [44]. These fluctuations are inherent to the streaming algorithm, which induces a fluctuation on the order of the temperature. In the current approach, an initial temperature is converted into an enthalpy value. It turns out that this conversion leads to big fluctuations in simulated variables, impacting the accuracy of the model severely.

It was found by Spek [25] that scaling and stretching the initial enthalpy values lead to a significant decrease of the impact of the fluctuations on the thermal field simulations. Firstly, scaling influences both the size of the enthalpy values and enthalpy fluctuations. Subsequently stretching the enthalpy spectrum reduces the relative contribution of the fluctuations on the overall field. Figure 3.3 shows how stretching and scaling impact the original enthalpy fluctuation.

To ensure accurate results, two requirements on enthalpy values have to be met. First of all it was observed that the the maximum enthalpy in the simulation should not exceed $\mathcal{O}(10^2)$. Higher values resulted in the appearance of fluctuations. Secondly, the maximum relative enthalpy difference must be large enough for the fluctuations to be insignificant. This is achieved by obeying the relation $\frac{(h_{max}-h_{min})}{h_{max}} \approx 1$.

Concretely, these two statements lead to a set of transformation rules for the sensible enthalpy, temperature and latent heat. These rules can be formulated as follows for the enthalpy and temperature

$$\hat{h} = (h - h_{min})\frac{\hat{h}_{max} - \hat{h}_{min}}{h_{max} - h_{min}} + \hat{h}_{min} \tag{3.22}$$

$$\hat{T} = (T - T_{min})\frac{\hat{h}_{max} - \hat{h}_{min}}{h_{max} - h_{min}} + \hat{T}_{min} \tag{3.23}$$

where a hat represents the transform. In this study $\hat{h}_{max}$ and $\hat{h}_{min}$ are chosen to be 1 and 0 respectively, but any choice that satisfies the aforementioned requirements would suffice. All enthalpy and temperature values within the simulations should be scaled according to equation 3.22 and equation 3.23.
The scaled minimum temperature $\hat{T}_{min}$ is also 0 for the current value of $\hat{h}_{min}$.
The transformed solidus temperature $\hat{T}_s$ and liquidus temperature $\hat{T}_l$ are defined as follows,

$$\hat{T}_s = \frac{h_s}{C_{p,s}} \tag{3.24}$$

$$\hat{T}_l = \hat{T}_s + \frac{\hat{h}_l - \hat{h}_s}{C_{p,m}} \tag{3.25}$$

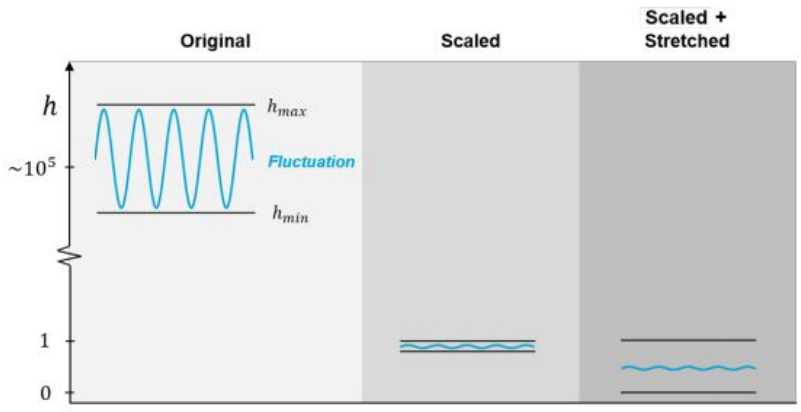


Figure 3.3: Depiction of the effect of scaling and stretching sensible enthalpy values on the enthalpy fluctuations. Here h_{max} and h_{min} correspond to the maximum and minimum enthalpy value within the simulation. Scaling the enthalpy reduces the absolute enthalpy fluctuation, while stretching reduces the relative impact it has on the simulation variable [25].

Lastly, the latent heat should also be transformed since it also represents an enthalpy contribution. Since the latent heat is defined as the difference between total enthalpy and sensible enthalpy, no offset value has to be used. Thus the latent heat can be formulated as

$$\hat{L} = L\frac{\hat{h}_{max} - \hat{h}_{min}}{h_{max} - h_{min}} \tag{3.26}$$

Before initialization takes place, all input values for enthalpy, temperature and the latent heat should be converted according to the aforementioned transformation rules. This ensures that the scaling and stretching procedure is effectively applied to the thermal distribution function, so that ensuring thermal fluctuations will not degrade the accuracy of the solutions.

3.2.4 Phase interface treatment

Within the freezing process, liquid fluid is transformed steadily into ice. As result the newly formed ice layer acts a solid wall, preventing liquid phase to flow into the solid phase. Consequently, at the phase interface a no-slip condition has to be imposed for the fluid behavior to be physically accurate. Within the mushy phase, fluid motion should be slowed down to account for the presence of the solid phase.

Noble and Torczynski [28] proposed the immersed boundary method (IBM) as a suitable approach to enforce the relevant physical effects at the phase interface. IBM relies on modifying the collision step, which implicitly keeps track of the solid-liquid interface. Within the FM-LBM each post-collision distribution is modified with a special collision operator

$$f_i^s\left(\mathbf{x} + \frac{\mathbf{c}_i\Delta t}{2}, t + \frac{\Delta t}{2}\right) = f_i\left(\mathbf{x} - \frac{\mathbf{c}_i\Delta t}{2}, t - \frac{\Delta t}{2}\right) + (1-B)\Omega_i + B\Omega_i^s; \tag{3.27}$$

where B is a collision constant that controls how dominant the special collision operator is over the regular collision operator. Within the ice ($f_l = 0$), B should have the value 1 to impose the no-slip condition via Ω_s. When the material is fully liquid on the other hand, normal collision should be applied and hence B should take the value 0 . Taking these considerations into account, B is defined as follows

$$B = \frac{(1-f_l)\epsilon}{(f_l + \epsilon)} \tag{3.28}$$

where ϵ is a small parameter that prevents B from division by zero. The collision term Ω_i is

$$\Omega_i^s = f_j\left(\mathbf{x} - \frac{\mathbf{c}_i\Delta t}{2}, t - \frac{\Delta t}{2}\right) - f_i\left(\mathbf{x} - \frac{\mathbf{c}_i\Delta t}{2}, t - \frac{\Delta t}{2}\right) + f_i^{\mathrm{eq}}(\rho, \mathbf{u_s}) - f_j^{\mathrm{eq}}(\rho, \mathbf{u}), \tag{3.29}$$

where $\mathbf{u}_s$ is the solids velocity. In principal, collision operator Ω_i^s "bounces back" the non-equilibrium part of the distribution function. However, when $\mathbf{u}_s = 0$ the above equation reduces to

$$\Omega_i^s = f_j\left(\mathbf{x} - \frac{\mathbf{c}_i \Delta t}{2}, t - \frac{\Delta t}{2}\right) - f_i\left(\mathbf{x} - \frac{\mathbf{c}_i \Delta t}{2}, t - \frac{\Delta t}{2}\right), \tag{3.30}$$

which corresponds to a regular zero-slip velocity condition. At the end of the FM-LBM collision step, collision operator Ω_i^s is determined and subsequently f_i^s is calculated. Now the distribution function f_i^s is the one that gets streamed to neighboring lattice sites instead of the regular post-collision function f_i.

The IBM has been successfully implemented in LBM simulations in multiple studies [27] [25] [26]. It was found that $\epsilon = 0.05$ gives satisfactory results and this value will be used in the rest of this research.

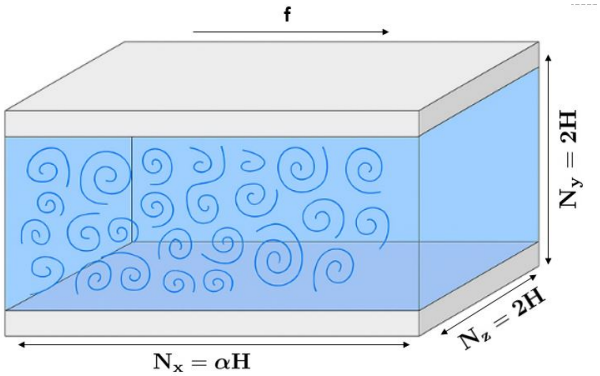


Figure 3.4: Schematic depiction of the channel being considered in this study. A body force f is applied along the streamwise direction. α is a variable scaling parameter.

3.3 BOUNDARY CONDITIONS

In order for a simulation to represent a realistic physical system, imposing boundary conditions at the edges of the simulation domain is crucial. Normally this boils down to prescribing a treatment for the simulation variables at the boundaries. However, in the LBM applying boundary conditions is more abstract since particle distributions are simulated rather than physical variables. The question of boundary conditions within the LBM is thus a question of prescribing a treatment for the missing distributions at the boundaries that realistically impose the desired values of the physical variables. The following sections will present all the boundary conditions that have been used in this study. A depiction of the channel setup considered in this study is shown in figure **??** so it is clear which situation applies to which boundary.

3.3.1 Wall conditions

All LBM boundary conditions being considered in this study are link-wise boundary conditions, since they are better suited at treating solid-liquid interfaces and are easy to implement [35]. As a consequence, the first node from the physical boundary needs to be placed at half a grid spacing. In this way, the boundary lies at the halfway mark of the lattice link between the boundary node (which is the first node in the computational domain) and the first solid node (which corresponds the first node in the solid region). Figure 3.5 shows a visual representation of this lattice links. Here the grey-shaded area corresponds to the solid region and the white area corresponds to the inner domain. To incorporate link-wise boundary conditions within the LB framework, an extra layer of nodes is added to the domain on all sides. These nodes are called ghost nodes and are used to keep values used for boundary treatment. The node at x_{N+1} is considered a ghost node in the current work.

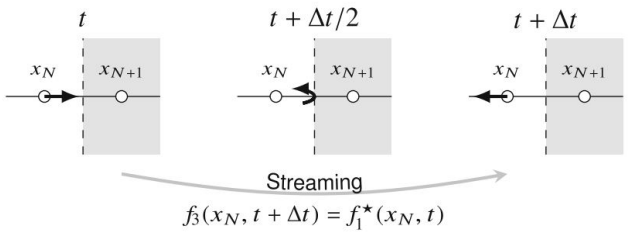


Figure 3.5: Schematic depiction of the halfway bounce-back (hbb) boundary condition a solid wal. [35]. Here the grey region corresponds to solid region and the white region to the interior. The line connection the two nodes is called a lattice-link.

At a solid boundary there can be no transport of momentum. As a result a no-slip velocity condition has to be enforced at the wall, which ensures zero velocity. To achieve this in LBM simulations ,the halfway bounce-back (hbb) method can be employed. Figure3.5 gives a visual explanation of the hbb boundary method and can be understood in terms of a traveling particle. Here a particles velocity at boundary node x_N points towards the solid node at position x_{N+1}. During the streaming step, this particle travels towards x_{N+1}, but is bounced back at the position of the solid boundary. Consequently, the particle travels back to the original position, but now posses the inverse velocity. To translate the effect of the solid wall to the case of particle distribution functions, the populations should be inversed at the boundary node during the collision step. As a result the half-way bounce back method can be imposed by

$$f_i(\mathbf{x}_b, t+\frac{\Delta t}{2}) = f^*(\mathbf{x}_b, t-\frac{\Delta t}{2}) - 2w_i\rho_w\frac{\mathbf{c}_i\mathbf{u}_w}{c_s^2} \tag{3.31}$$

where $\mathbf{x}_b$ represent the position of the boundary node, f^* is the inverse value of f, ρ_w is the density of the fluid near the wall and $\mathbf{u}_w$ is the velocity of the wall. For a non-moving solid wall, $\mathbf{u}_w = 0$ simplifying the equation.

Imposing a boundary value that is non-zero creates another challenge. In the current grid, boundary nodes do not coincide with the boundary itself. As result, it is not possible to directly enforce a temperature value here. To be able to prescribe the right temperature value to the wall, the boundary has to communicate the value to the boundary node, A suitable way to do this is via an anti-bounce back scheme [45]

$$g(\mathbf{x}_b, t+\frac{\Delta t}{2}) = -g^*(\mathbf{x}_b, t) + 2w_i h_w \tag{3.32}$$

where the wall temperature can be imposed via the sensible enthalpy h_w. Here the asterisk once again denotes the inverse of the distribution function.

3.3.2 Inlet- and outlet conditions

In channel flow applications it is needed to prescribe suitable boundary conditions for the fluid at the inlet and outlet. The easiest and least computationally expensive way to do this is by using periodic boundary conditions (PBC). In this condition it is assumed that the outflow re-enters the domain at the inflow, essentially simulating an infinitely long dimension. Periodic boundary conditions for channel flows are suitable for open boundaries in the x- and z-directions. In order to implement periodic boundary conditions in the LB algorithm, the domain needs a layer of ghost nodes For a channel with length L along the periodic dimensions, periodic boundary conditions are defined as

$$f_i\left(\frac{\Delta x}{2}, t-\frac{\Delta t}{t}\right) = f_i\left(L-\frac{3\Delta x}{2}, t+\frac{\Delta t}{2}\right), \tag{3.33}$$

$$f_i\left(L-\frac{\Delta x}{2}, t-\frac{\Delta t}{t}\right) = f_i\left(\frac{3\Delta x}{2}, t+\frac{\Delta t}{2}\right). \tag{3.34}$$

The first equation copies the pre-collision distribution function values at the outlet to the first ghost node. During the propagation step, the distributions pointing into the domain are streamed to the first node in the domain. Similarly, the second equation stores the values from the inlet at the ghost node at the outlet.

Even though the PBC is a useful tool to simulate fully developed channel flow, it does not represent a finite system. Non-physical correlations in turbulence structures can arise due to eddies being artificially repeated instead of evolving naturally [13]. To realistically simulate a developing ice layer, the flow within the channel should develop a natural turbulent flow profile. This can be done by extracting real-time turbulence flow data from a concurrent simulation [29]. Figure 3.6 shows how the two domains should be setup. The concurrent domain is shown on the left, where periodic boundary conditions are applied in the streamwise direction. The shaded plane corresponds to the sampling plane from which turbulent flow data is extracted. In turn this data is used as an inflow profile for the main simulation. For both domains it is crucial that grid spacings and Reynolds number should match to arrive at accurate results [46]. Besides that, it is convenient to use the same dimensions for the inlet plane of the main simulation and the extraction plane. This method of generating a turbulent inflow profile is also referred to as the temporal strong recycling method.
The procedure of implementing the inflow condition in the LBM is straightforward. After streaming, the distribution functions associated with the extraction plane are copied to first layer of nodes in the simulation domain associated with $x = 1$.

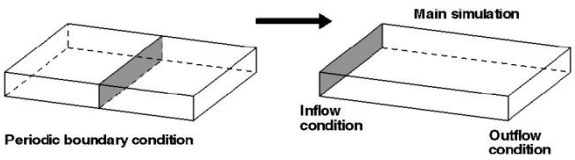


Figure 3.6: Visual representation of the concurrent domain and the main simulation domain [29]

Imposing a realistic thermal inflow can be easily implemented by applying the anti-bounce back method described in equation 3.32 on the inlet plane. A suitable inflow temperature can be set by declaring a desired enthalpy value h_w.

At the end of the channel the thermal fluid should flow out of the domain smoothly, without causing numerical errors. To this end, a zero gradient Neumann condition is conidered, which enforces a zero-gradient at the boundary [47]. To implement this boundary condition in the LBM simulations, the distribution functions for the plane $N-1$ at the boundary node near the outlet is copied to the plane layer N at the neighboring ghost node:

$$f_i\left(N, t-\frac{\Delta t}{2}\right) = f_i\left(N-1, t-\frac{\Delta t}{2}\right) \tag{3.35}$$

$$g_i\left(N, t-\frac{\Delta t}{2}\right) = g_i\left(N-1, t-\frac{\Delta t}{2}\right) \tag{3.36}$$

This effectively sets the velocities and temperatures near the outlet to $\mathbf{u}(N-1,t) \approx \mathbf{u}(N,t)$ and $T(N-1,t) \approx T(N,t)$.

3.4 GPU IMPLEMENTATION

In this thesis, the DDF FMLBM model is implemented to be compatible with parallel programming using the CUDA framework. The LBM method contains inherently local and simple operations, making the perfect match for parallelization using a GPU implementation. The main reason for parallelization is the potential speed-ups gained from switching from a CPU-based model to a GPU-based one. This section will discuss the specific numerical implementation of the of the freezing model within the CUDA.jl framework.

3.4.1 Julia

The FM-LBM code in this thesis is implemented using the programming language Julia. Julia is a relatively new programming language, which is designed for high-performance numerical and scientific computing [48]. It offers the convenience of a high-level languages like Python and Matlab, but with the speed and efficiency of compiled languages like C. This performance is achieved through Julia's unique just-in-time(JIT) compilation via the LLVM framework. Of particular interest to this study is the fac that Julia offers more support for parallelism than Python does. For example, Julia allows programmers to explicitly manage GPU memory, including the allocation of shared memory resources.

Julia supports the use of different GPU backends through packages as CUDA.jl for NVIDIA GPU's and AMD.jl for AMD-powered GPU's. In this study, an GPU application is made that can run on multiple backends. This versatility is achieved through the Julia package KernelAbstractions.jl, which allows kernel code to be written in a generic and backend-independent way. For instance, if an NVIDIA GPU is detected, the kernel is executed using the CUDA.jl backend.

The specifics of the GPU-implementations in this study closely resembles the methodology described by Entes [49]. A brief summary of the key implementation steps, which are also applied in this thesis, is provided below. For more specifics, the reader is referred to the thesis of Entes.

3.4.2 Computational workflow and Kernel Design

Kernel functions are defined for the collision step, the propagation step, and the boundary treatment. When constructing these kernels, it is important to keep in mind that a maximum of three dimensions can be indexed in CUDA. This limits the amount of velocity directions and spatial domains that can be parallelized. The collision kernels are only parallelized along the spatial dimensions, due to the fact that the matrix multiplication uses information from all the different velocities at one location of the distribution function. The propagation kernel is parallelized however across both the velocity and spatial dimensions, due to streaming step calculations being symmetrical along different velocity directions.

In contrast to the study of Entes [49], this study uses multiple kernel functions for the boundary treatment. Kernels involving open boundaries are parallelized along the velocity direction and the spatial plane dimensions perpendicular to the direction they are applied to. Furthermore, wall conditions involve only one spatial dimension and is thus easily parallelized.
In addition to the aforementioned kernels, a functionality involving saving running simulation statistics on the GPU was implemented in this study. This kernel only involves all the three spatial dimensions. Furthermore, the saving is not called in every iteration of the LBM algorithm. These kernels only involve all the three spatial dimensions. Lastly an extra kernel is present in this study to update the liquid fraction, which is parallelized over three-dimensional space.

Before the start of the simulation, input data and arrays are initialized into Julia structs. These structs and its data are then transferred from CPU memory to GPU memory, ready to be used by GPU kernels. After that, the distribution functions are initialized on the GPU. When the necessary structs are fully initialized, the main LBM simulation can start. In the code, this corresponds to a for loop that runs N_T amount of iterations of the following sequence of kernels:

1. Liquid-fraction kernel
2. Wall-boundary condition kernel

3. Collision kernels
4. Propagation kernel
5. Open-boundary conditions kernel
6. Save kernel (when needed)

The above procedure is stopped when the iteration number t exceeds the total number of time steps specified. The last steps involve GPU data being transferred back to the host for saving.

3.4.3 Race conditions

Similar to Entes, care is taken to prevent race conditions using the propagation kernel. When a single distribution function is used for both reading and writing within a kernel, it is possible that threads read from a memory location that is already updated, consequently writing the wrong value to a lattice node. To prevent this race condition from occurring, two distribution functions are used per physical field. Specifically in the case of flow field simulations these distribution functions are denoted by f_{pre} and f_{post}. Now threads read from f_{pre} and write to the f_{post}. Furthermore, synchronization of threads is performed by using CUDA.@synchronize upon completion of a kernel.

3.4.4 Memory coalescence

To achieve high parallel efficiency, it is important for threads executed in the same block to access data locations that are close to each other, reducing the amount of time threads need on average to read memory locations. Two important steps are taken to achieve this memory coalescence. Firstly, 4D arrays corresponding to the distribution functions and the 3D arrays corresponding to macroscopic variables are converted to 1D arrays. On top of that, it is important that in the 1D arrays neighboring grid points have the same discrete velocity. Concretely, 1D arrays are indexed in the following way [49]

$$\text{3D to 1D conversion:} \quad idx = x + yN_x + zN_xN_y, \tag{3.37}$$

$$\text{4D to 1D conversion:} \quad idx = x + yN_x + zN_xN_y + qN_xN_yN_z. \tag{3.38}$$

Here (x, y, z) corresponds to spatial coordinates, while q represents the index of the discrete velocity. This layout ensures improved computational efficiency, especially during the matrix multiplication performed in the collision kernel.

3.4.5 Shared memory and matrix multiplication

Julia enables the assignment of variables to different levels of the GPU memory hierarchy, discussed in Section 2.5. This enables usage of the resource of shared memory, which is a fast memory type close to the CUDA core. In the current model, shared memory is only useful in the collision kernel, due to threads within a block being able to read and write to the same memory locations. This fact makes it hard to use in a kernel which also uses memory locations from neighboring grid nodes. Within the kernel, two nested $q \times q$ for loop are present to perform the filter-matrix multiplication that retrieves either the solution vector α_k or the post-collision distribution function. Arrays in these for loops are frequently accessed, so the effect of faster memory accesses is most heavily felt here. Therefore the solution vector is put into shared memory. Building on this, further performance gains are achieved by choosing a smart way of indexing arrays within the for loops. The inner loop index is iterated q^2 times, while the outer loop is iterated just q times. Therefore, the loop indexing is chosen in such a that the solution vector stored in shared memory is accessed q^2 times in each loop, while the distribution function stored in global memory is only accessed q times.By doing this, global memory accesses are minimized, while shared memory accesses are maximized.

Further minor improvements

Another measure taken to improve computational speed on the GPU is by using 32-bits integer and float values. Furthermore, number of threads per block are always chosen to be a multiple of 32 threads, so all threads in a warp are fully utilized.

3.5 SIMULATION REQUIREMENTS

3.5.1 Direct Numerical Simulation

For a turbulent simulation to be called a DNS, strict requirements have to be met so as to capture all relevant information contained in the flow. As discussed in Section 2.4.4, the grid size should be small enough to capture the smallest turbulent scales, while the domain should be able to accommodate the largest eddies within the flow. The small-scale dynamics in the flow system is determined by the Kolmogorov length scale. To faithfully reproduce the effects of the small-scales, it is sufficient to have a step size on the order of the Kolmogorov length [13]. In turbulent channel flow simulations, the Kolmogorov length is a function of the distance to the boundary i.e. $\eta = \eta(y)$, where the smaller values are expected closer to the wall. [This needs a reference] It can be derived that the smallest eddy is the size of roughly two wall units $\eta^+ \approx 2$ [50, 51]. The grid cell within the LBM simulation is uniform in all directions, thus the all-resolving step size should be set at Δx^+.
The adequacy of the extend of the spatial domain depends on the type of boundary that is applied along a certain dimension. In the wall-normal direction a non-homogeneous no-slip condition is used to enforce the presence of a solid wall. As a result, the boundary inhibits the growth of an eddy that is larger than the channels diameter. Consequently, the extend of the y-direction will always be sufficient to capture the large eddy behavior [13]. The stream- and spanwise directions are defined to be infinite in length in channel flows. To mimic this behavior for a finite computational box, periodic boundary conditions are imposed along these directions. Now fluid flowing out is reintroduced at the beginning of the channel. When the size of the periodic directions is not sufficient to accommodate the largest eddies, the turbulent structure will overlap with its own image. Consequently, information of these large-scale eddies is lost. Accurate simulation of the largest scale structures, thus relies on the spatial correlation to drop to zero within half the length of the channel [13]. From the DNS of turbulent channel flow performed in the study of Kim et al. [51] , spatial correlation in the streamwise- and spanwise direction drops to zero at roughly $4H$ and $2H$, respectively. A strict DNS should therefore have a minimum spatial extend of $L_x \times L_y \times L_z = 8H \times 2H \times 4H$.

When interested in lower-order statistics, like the mean velocity and Reynolds stress terms, a smaller computational box can be sufficient to capture the essential physics [52]. Flores et al. suggest that the minimal box $L_x \times L_y \times L_z = 6H \times 2H \times 3H$ for $Re_\tau = 180$ is sufficient.

Unlike periodic boundary conditions, realistic turbulent inflow- and outflow conditions allow turbulent structures to evolve naturally in the streamwise direction without artificially constraining the large-scale eddies. As a result, no minimum domain length is required to accurately capture eddy evolution along this direction. However, care must be taken that the inlet turbulent profile is consistent with flow developing within the domain. To achieve this, both the auxiliary- and main domain are run simultaneously for several channel flow-through times t_f before collecting information from the model. In this way a physical consistent inflow profile is established for the main simulation. Furthermore, the same grid resolution, cross-sectional dimensions ($N_y \times N_z$), Reynolds number, and physical parameters are chosen for the auxiliary domain to ensure consistent turbulent profiles are generated in both domains.

In LBM simulations that aim to numerically reproduce the solutions of the the incompressible Navier-Stokes equations, the step size and the timestep are intricately related through the Mach number requirement $Ma << 1$. Specifically, an increase in timestep results increasing the LB velocity and thus its Mach number. Therefore, choosing a step size equivalent to the Kolmogorov length automatically determines the maximum stable timestep. Resolving all relevant temporal scales is thus automatically satisfied when the Kolmogorov length is chosen as the grid resolution.

To retrieve reliable flow statistics from a turbulent thermal flow, the simulation goes through three different simulation phases. The first phase consists of initializing the correct physical fields, after which the simulation is started. Initialization of numerical data typically induces a transient phase during which turbulent flow is not yet stationary. The second phase thus consists of running the simulation until its statistical properties are converged. After convergence, statistical data can safely be retrieved, which comprises the third phase.

3.5.2 Initialization

In turbulent flow simulation, the accuracy of the simulation depends strongly on the initial condition, because of the presence of the non-linear term in the Navier-Stokes equations. Small errors in the initial state can propagate over time and can seriously affect the accuracy of the simulation [35]. Consequently, care must be taken in choosing a suitable initial flow field that accurately represents the target turbulence characteristics of the flow field. Initial turbulent flow field data from numerical experiments of [24] is used in this thesis. Table 3.1 lists the numerical settings for flow fields that were used in this study.

Table 3.1: Numerical settings of initial flow datasets used in this study [24].

Name	Re_τ	Re_m	$N_x \times N_y \times N_z$	u^+	Δy^+	ν	g
Short domain	180	5590	256 × 128 × 128	6.67e-3	2.8	2.37e-3	6.94e-7
Long domain	180	5590	512 × 128 × 128	6.67e-3	2.8	2.37e-3	6.94e-7

[Check if body force g is a term that is used more often] The initial flow profiles were generated using a DNS of perturbed laminar fields, which evolved into sustained turbulence. From the table it is clear that the spatial requirements discussed in Section 3.5.1 are not met. However, the low-order turbulence statistics of this dataset show satisfactory correspondence to benchmark studies [24]. Only the RMS velocity flu-cations and vorticity fluctuations are reported to be overestimated by this configuration.

Translating the flow variables of density and velocity to distribution function values is straightforward within the FMLBM-method. The flow variables can be inserted into the pre-collision solution vector α^- defined in Eq.3.12, and subsequently be translated into the distribution function f using Eq. 3.9.

In this study the grid will be refined in order to concur with the resolution requirement of $\Delta y^+ = \eta^+$. Consequently, variables from the initial dataset have to be interpolated onto the new, refined grid. In Julia, three-dimensional interpolation can be straightforwardly implemented using the Interpolations.jl package. Cubic B-Spline interpolation is performed on the initial density and velocity arrays to create smooth interpolated values at the new grid locations.

3.5.3 Convergence

Retrieving meaningful statistical data that the flow has reached a sufficiently stead-state situation. Without this convergence, any computed statistics may be influenced by transient behavior, which leads to incorrect representation of the flow dynamics. One common approach to quantify a simulations convergence is by quantifying the change of a certain flow variable in time. convergence to steady-state is by employing the relative L_2 error norm. In this thesis, the spatially average velocity mean is tracked over time and its deviation from a previous iteration is computed via:

$$L_2^{conv}(t) = \sqrt{\frac{\sum_{\mathbf{x}} \left[\overline{u}(\mathbf{x}, t) - \overline{u}(\mathbf{x}, t - \Delta t)\right]^2}{\sum_{\mathbf{x}} \overline{u}^2(\mathbf{x}, t - \Delta t)}}. \tag{3.39}$$

Here, the summation is performed over the entire three-dimensional domain. The Root Mean Square (RMS) difference between successive realizations is normalized, providing a consistent metric to compare the relative change of the solution over time. When the calculated L_2^{conv} term falls below a chosen threshold, the simulation is considered to be sufficiently converged for retrieving turbulence statistics. It is reported that the convergence threshold is $L_2 \leq 1 \times 10^{-3}$ sufficiently acceptable for simulations that employ 32-bit floating-point precision.

3.5.4 Averaging window and sampling rate

Care needs to be taken when collecting statistical data. To produce accurate statistical results, enough non-correlated data points should be included in the statistical analysis. Furthermore, the sample rate should be chosen as to retrieve enough data points. The study by Vinuesa et al. [53] provides useful guidelines for selecting the appropriate averaging windows and rate of sampling. To this end, the eddy turnover time(ETT) is defined as

$$ETT = \frac{tu_\tau}{H}, \tag{3.40}$$

where t is the simulation time. In this formula time is normalized by the typical advection time of the biggest scales defined by u_τ / H. ETT is a non-dimensional parameter that quantifies how many times the large scales have traveled distances equal to the channel's half-height. This is a universal parameter which gives information on have many times the flow in the channel is refreshed, independent of its flow configuration. It is therefore a robust parameter to check whether enough cycles of advection are used in the averaging window, to ensure uncorrelated data points. To also account for differences in computational domain sizes, a normalized eddy turnover time is defined as follows.

$$\mathrm{ETT}^*_{\mathrm{channel}} = \mathrm{ETT}_{\mathrm{channel}} \frac{L_x L_z}{L_{x,\mathrm{min}} L_{z,\mathrm{min}}}, \tag{3.41}$$

where $L_{x,min} = 6H$ and $L_{z,min} = 3H$ are the minimal streamwise and spanwise lengths previously mentioned in section 3.5.1. Furthermore, a sufficient sample rate is found to be equal to [53],

$$\Delta t^+ = 17 \tag{3.42}$$

4 VALIDATION OF TURBULENCE MODEL

The primary aim of this thesis is creating a realistic and computationally effective simulation tool that can model freezing of salt flow under turbulent conditions within a cooled MSFR heat exchanger. To this end, a GPU-accelerated FM-LBM turbulence model has been created, and its accuracy in simulating turbulent flow behavior must therefore be validated. First of all, an overview of the physical turbulence model and the numerical setup is provided in section 4.1. section 4.2 discusses the literature studies selected to assess the model's validity, alongside giving an overview of simulations performed. Furthermore, section 4.3 contains the results of periodic simulations using different domain sizes and grid resolutions. In section 4.4, non-periodic inflow- and outflow conditions are added mimicking a more physically accurate situation. Furthrmore, he effectiveness of the GPU-implementation is discussed in section 4.5. section 4.6 gives a short conclusion of this chapter.

4.1 COMPUTATIONAL SETUP

The GPU-accelerated FM-LBM fluid dynamics model is developed to simulate correct turbulence solutions from the Navier-Stokes equation in channel flow. A schematic depiction of the physical problem being investigated is given in Figure **??**. In this picture the number of grid points in the streamwise, wall-normal and spanwise directions are indicated with N_x, N_y and N_z respectively. The ratio parameter α can take either the value 1 or 2. A body force $\boldsymbol{f}$ is applied along the streamwise direction, forcing the fluid through the channel. In all simulations, a halfway bounce-back condition is applied in the wall-normal direction to impose a no-slip condition. Furthermore, periodic boundary conditions are applied along the spanwise direction. Different computational domain sizes, grid resolutions, and streamwise boundary conditions are used across the simulations in this section. The specific configurations of each simulation are detailed in their respective sections.

To simplify comparison with literature studies, the fluids properties are chosen to match those of water. An overview of all the relevant physical input parameters in this study, is given in Table 4.1

Table 4.1: Physical input parameters used in turbulent simulations in this section. The kinematic viscosity and density are representative of water at room temperature.

Quantity	**Symbol**	**Value (SI)**
Channel dimensions	$L \times 2H \times W$	$0.10 \times 0.05 \times 0.05$ m
Kinematic viscosity	ν	$1.7 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$
Body force	f_x	$5.99 \times 10^{-3}\,\mathrm{m\,s^{-2}}$
Input density	ρ_0	$1000\,\mathrm{kg\,m^{-3}}$
Wall shear velocity	u_τ	$1.12 \times 10^{-2}\,\mathrm{m\,s^{-1}}$

4.2 BENCHMARK STUDIES AND SIMULATION OVERVIEW

An overview of all the simulations performed in this study is presented in Table 4.2. In total four simulations are studied using periodic boundary conditions in the streamwise directions. These simulations are denoted in the table by the abbreviation PBCx. The difference between these periodic simulations are the aspect ratio α of the domain length and the applied grid resolutions. Specifically, the number of nodes in the x-direction can vary between $N_x = 4H$ and $N_x = 8H$. For both domain size, two different grid resolutions ($\Delta^+ = 2.8$ and $\Delta^+ = 2.0$) totaling four distinct periodic simulations. Furthermore, flow parameters from Table 4.1 are provided in LB units. The three last columns detail the averaging period and the sampling interval needed to meet a sampling resolution similar to the benchmark study of Kim et al. [50] [53]. This averaging period is normalized by the eddy-turnover time (ETT). Furthermore, section 4.4 presents the results of the realistic inflow scenario, which is called IBC in the table. The streamwise inflow condition is provided by the strong recycling method, while the streamwise outflow condition is imposed by a zero-gradient Neumann condition. The strong recycling method relies on running an auxiliary channel flow simulation in tandem with the main simulation. Numerical input parameters for the IBC simulation are the same as in table 4.1.

Table 4.2: Overview of physical parameters in LB units and parameters used for averaging. The wall shear velocity and body force were adjusted to ensure $Re_\tau = 180$. The sampling interval indicates the number of time steps between successive saves.

Simulation	$N_x \times N_y \times N_z$	Δ^+	ρ	ν	f_x	u_τ	Averaging period (ETT)	Total timesteps	Sampling interval (timesteps)
PBC1	$256 \times 128 \times 128$	2.8	1.0	2.37×10^{-3}	6.94×10^{-7}	6.67×10^{-3}	112.5	1.19×10^6	1100
PBC2	$368 \times 184 \times 184$	2.0	1.0	2.37×10^{-3}	4.83×10^{-7}	6.67×10^{-3}	56.2	1.61×10^6	1500
PBC3	$512 \times 128 \times 128$	2.8	1.0	2.37×10^{-3}	6.94×10^{-7}	6.67×10^{-3}	112.5	6.17×10^5	1100
PBC4	$736 \times 184 \times 184$	2.0	1.0	2.37×10^{-3}	4.83×10^{-7}	6.67×10^{-3}	56.2	8.87×10^5	1500
IBC	$512 \times 128 \times 128$	2.8	1.0	2.37×10^{-3}	6.94×10^{-7}	6.67×10^{-3}	112.5	6.17×10^5	1100

To validate the FM-LBM turbulence model, the results are compared against DNS data from two benchmark studies. The first successful DNS of turbulent channel flow using at $Re_\tau = 180$ has been performed by Kim, Moin and Moser (KMM) in 1987 [51]. Later, this DNS data was made open-source based on the paper published in 1999 [50]. Periodic boundary conditions are applied in streamwise- and spanwise directions, while a no-slip is enforced near the wall. The kolmogorov length scales with distance to the wall, so to balance accuracy and speed a non-uniform wall normal spacing was employed. This grid spacing in the wall-normal was defined by

$$y_j^+ = z\frac{u_\tau}{\nu} = 180\left(1 - \cos\left(\frac{(j-1)\pi}{N_y - 1}\right)\right), \tag{4.1}$$

where was $N_y = 129$. This resulted in a grid spacing of $\Delta y^+ = 0.05$ near the wall and a maximum spacing of $\Delta y^+ = 4.4$ at center of the channel. Grid spacing in the streamwise- and spanwise directions were $\Delta x^+ \approx 12.7$ and $\Delta z^+ \approx 7$ respectively. Furthermore, the spatial extent of the computational domain was taken to be $4\pi H \times 2H \times \frac{4}{3}\pi H$, which meets the requirements discussed by Flores et al. [52]. Due to its reliable computational setup and publicly available data, the DNS of turbulent channel flow conducted by KMM has become a widely adopted benchmark in the literature for flows with $Re_\tau = 180$. [Add here citation to studies that use KMM as benchmark]

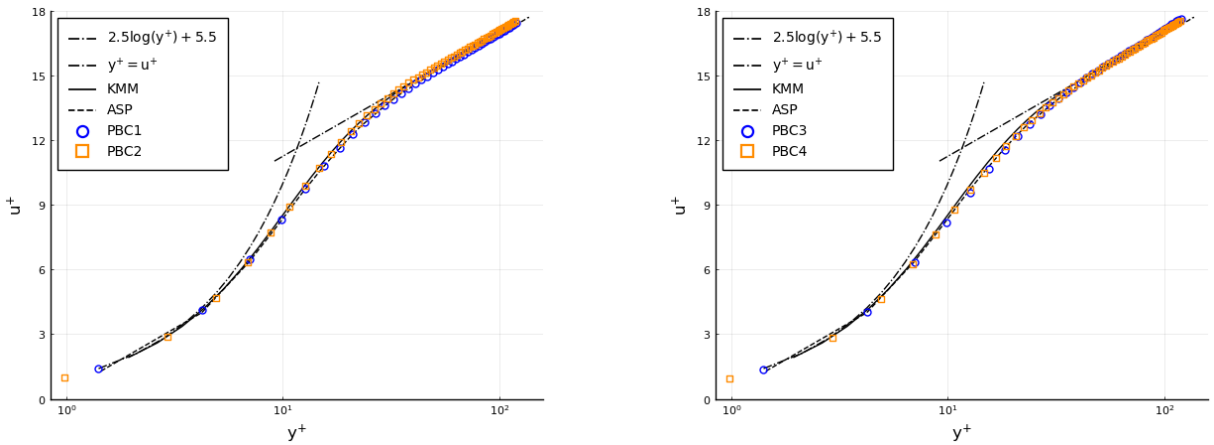
Another useful benchmark study is the one conducted by Amati et al. [54](ASP). A turbulent channel flow at $Re_\tau = 180$ has been simulated using a LBM numerical technique with a BGK collision operator. A uniform grid spacing of $\Delta^+ = 2.8$, which is similar to the spacing used in this study. This makes it interesting to use data from ASP as a benchmark. Once again, periodic boundary conditions have been used in the stream- and span-wise directions and a zero-slip in the wall-normal direction. Of these two benchmark studies, the one conducted by Kim et al. is considered to be the primary reference due to its more than adequate spatial extent and very fine wall-normal grid spacing.

Table 4.3: Overview of benchmark studies and simulations on turbulent channel flows at $Re_\tau = 180$ used in this section. Data from Kim et al.(KMM) [50] and Amati et al.(ASP) [54] are used to compare data from current simulations.

Simulation	Numerical method	Collision model	$N_x \times N_y \times N_z$	Δy^+	streamwise BC
KMM	Spectral	-	192 × 129 × 160	0.05 – 4.4	Periodic
ASP	LBM	BGK	256 × 128 × 128	2.8	Periodic
PBC1 (This study)	LBM	FMLBM	256 × 128 × 128	2.8	Periodic
PBC2 (This study)	LBM	FMLBM	368 × 184 × 184	2.0	Periodic
PBC3 (This study)	LBM	FMLBM	512 × 128 × 128	2.8	Periodic
PBC4 (This study)	LBM	FMLBM	736 × 184 × 184	2.0	Periodic
IBC (This study)	LBM	FMLBM	512 x 128 x 128	2.8	Strong recycling/Neumann

4.3 PERIODIC SIMULATIONS

Turbulence statistics obtained from the FMLBM turbulence model using periodic boundary conditions will now be compared to aforementioned benchmark studies. Relevant quantities that are considered are the mean stream-wise velocity, the RMS velocity fluctuations and the Reynolds stress. Data from PBC1 and PBC2, corresponding to grids using streamwise length $L_x = 4H$ are put into the same graphs. Similarly, simulations PBC3 and PBC4 using $L_x = 8H$ are plotted together. For simulations employing the same uniform grid size $\Delta^+ = 2.8$ as the initial flow fields, convergence to a statistically steady-state was immediately observed. However, simulations that used a refined grid as compared to the initial dataset needed at least 1 t_c to converge. To stay on the safe side, these simulations were first run for one full flow through time t_f before obtaining statistics.



(a) Simulation using domain length $L_x = 4H$

(b) Simulation using domain length $L_x = 8H$

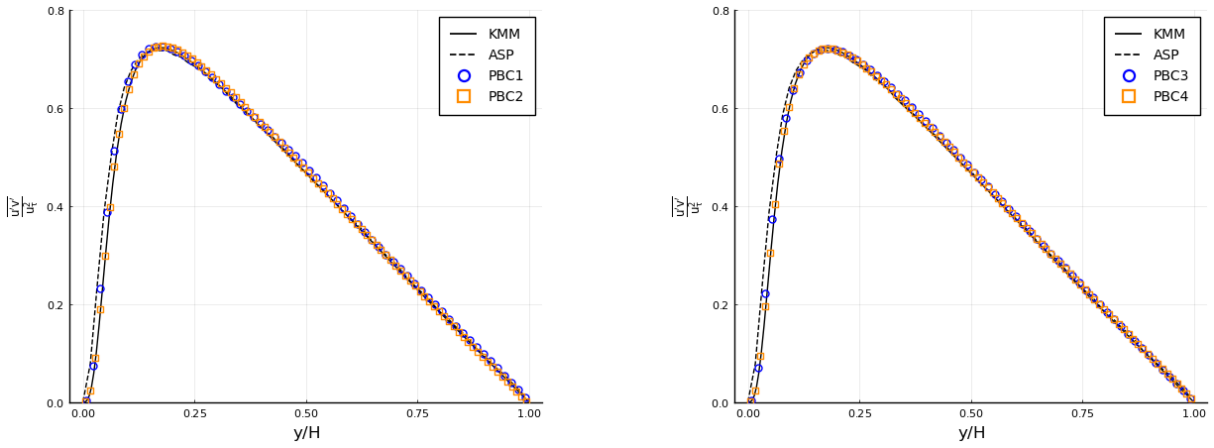
Figure 4.1: Graphs showing the mean streamwise velocity profiles near the wall expressed in wall units for $Re_\tau = 180$ as a function of wall distance expressed in wall units . In graph (a) streamwise length $L_x = 4H$ and in (b) $L_x = 8H$. The data points represented by circles have a uniform grid spacing $\Delta^+ = 2.8$, while rectangles correspond to $\Delta^+ = 2.0$. Additionally, analytical near-wall velocity profiles are included for comparison. The black lines correspond to data obtained from Kim et al. (KMM) [50] and Amati et al. (ASP) [54].

Mean Velocity

The mean-streamwise wall velocity profiles are shown in Figure 4.1 as function of wall distance expressed units. In the plot on the left PBC1 and PBC2 are displayed, which corresponds to the grid setup with $L_x = 4H$. The blue data points correspond to a uniform grid resolution of $\Delta^+ = 2.8$, while the orange data points originate from a grid with $\Delta^+ = 2.0$. Analytical expressions for near-wall turbulent velocity profiles are plotted by a black dash-dotted line. Additionally, the benchmark studies are distinguished by the solid black line (KMM) and the black dashed line (ASP). The plot in Figure 4.1b contains the data points of simulations PBC3 and PBC4,which correspond to a grid configuration using $L_x = 8H$. The different grid resolutions are represented by the same colors as in the left graph. The first observation to be made is that PBC1 exactly follows the line corresponding to data from ASP. Unsurprisingly, these two simulations used the exact same numerical configurations and thus the FMLBM model can simulate the mean-velocity as accurate as LBM literature studies. Furthermore, all the profiles follow the benchmark of KMM and the analytical solutions closely. Upon comparison, the blue and orange scatter points in both plots show a marginal shift towards the data from KMM, indicating a slight increase in accuracy when refining the grid. The impact of the grid length size can be derived by comparison between graphs. Data points from PBC1 and PBC2 are on overall slightly closer to the benchmark lines than scatter points from PBC3 and PBC4.

Reynolds Stress

The effectiveness of turbulent momentum transport in the wall-normal direction is the described by the Reynolds stress profiles. For FM-LBM model to succesfully correspond to characteristic turbulent flow at $Re_\tau = 180$, Reynolds Stress profiles should closely match benchmark studies. Figure 4.2 shows the retrieved Reynolds Stress profiles in a similar format as the mean-streamwise velocity results. From the graphs it is evident that all grid configurations capture the Reynolds stress almost perfectly. A slightly better correspondence is achieved by refining the grid.



(a) Simulation using domain length $L_x = 4H$

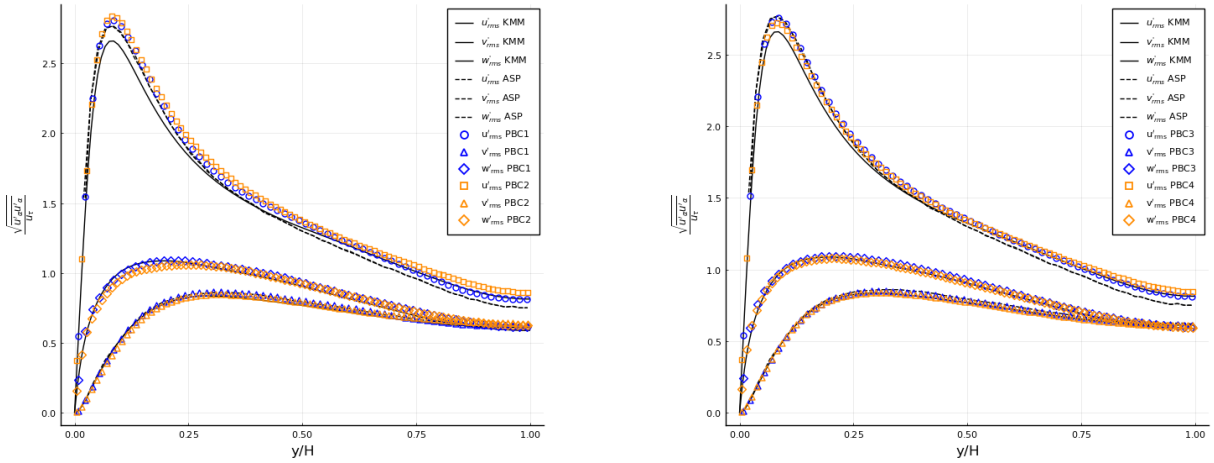
(b) Simulation using domain length $L_x = 8H$

Figure 4.2: Graph showing the results for retrieving the Reynolds Stress profiles. Figure (a) corresponds to grid configuration using $L_x = 4H$ while Figure (b) uses $L_x = 8H$. Furthermore, blue points indicate a grid size $\Delta^+ = 2.8$ is used, while blue points represents $\Delta^+ = 2.0$. The solid and dashed lines represent results from Kim et al. [50] and Amati et al. [54] respectively.

Root-Mean Square Velocity Fluctuations

Figure 4.3 shows the root mean square (RMS) velocity fluctuations u'_{rms}, v'_{rms} and w'_{rms}. . It is observed that the the simulation reproduces the general shape of the characteristic profile. In the buffer layer ($5 < y^+ < 30$) and the lower part of the logarithmic sublayer, turbulence instensities typically reach their peak values. This occurs because the shearing effect of the wall is still active, while the viscous damping is reduced. Within the plots, aforementiond region is approximately located at $0.02 < y/H < 0.2$. This coincides with the peaks of the u'_{rms} and w'_{rms} profiles. In general, the curves representing the fluctuations in the y- and z directions match the benchmark data accurately. The only exception are the data points from the PBC2 simulation, which shows a slight dip below the peak intensity of w'_{rms}. However, this dip is not observed for the curve from the PBC4 simulation.

Furthermore, all curves show an overshoot of the peak u'_{rms} value. The dashed line shows the u'_{rms} profile produced by ASP also suffers from a similar discrepancy. Throughout LBM literature similar results are published [55], [22] Amati et al.[54] argues that the overshoot likely stems from the LBM being only second-order accurate in time an space, while the study of Kim et al. uses a higher numerical technique [50]. When comparing the simulations based on domain length, it is clear that in simulations PBC3 and PBC4 the overshoot is reduced. This observation suggests that increasing the domain length more faithfully captures the turbulence effects within the flow. Such a result is expected as Flores et al. [52] argues that the current domain size is too small to capture all relevant large eddies. Using a larger domain length results in the capture of information that would otherwise be lost when employing a shorter domain. The effects of using a finer grid are ambiguous. In Figure 4.3a results of the finer grid for u'_{rms} deviate more from the benchmark than its coarser counterpart. Overestimation is especially visible near the center channel. As mentioned earlier, w'_{rms} significantly deviates around its peak value for the finer grid in PBC2. Inspecting the data points from PBC4 on the other hand, suggests that a finer grid resolution slightly reproduces the peak u'_{rms} better than its coarser counterpart. However, near the center channel a slight overestimation is present as well. Thus, it can be concluded that the effect of increasing the grid resolution beyond that of the initial dataset does not yield significant effects on accuracy. To save unnecessary computational costs, a grid size of $\Delta^+ = 2.8$ will be assumed for further simulations.



(a) Simulation using domain length $L_x = 4H$

(b) Simulation using domain length $L_x = 8H$

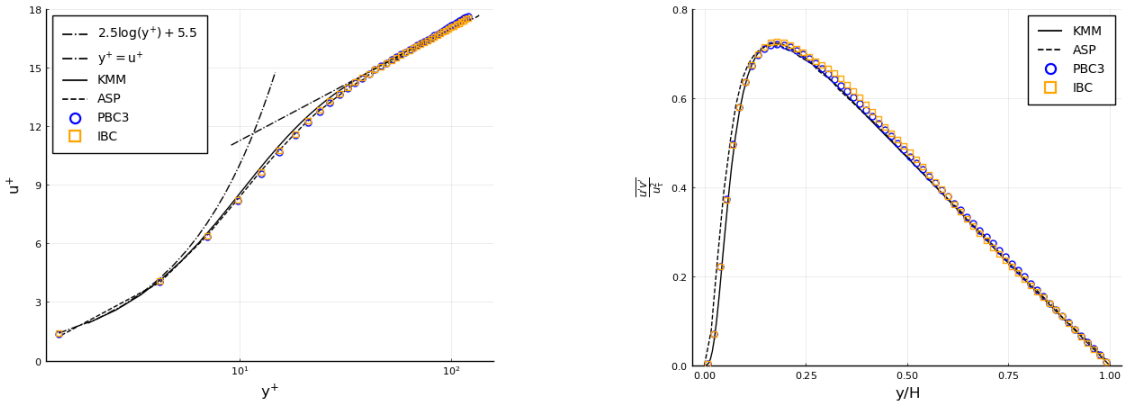
Figure 4.3: Figure showing the RMS velocity fluctations profile retrieved from FM-LBM model as function of the wall-normal distance. Figure (a) shows simulations using domain length $L_x = 4H$, Figure (b) shows simulations using domain length $L_x = 8H$.Furthermore, blue points indicate a grid size $\Delta^+ = 2.8$ is used, while blue points represents $\Delta^+ = 2.0$. The black lines correspond to data obtained from Kim et al. (KMM) [50] and Amati et al. (ASP) [54].

-

4.4 REALISTIC INFLOW SIMULATION

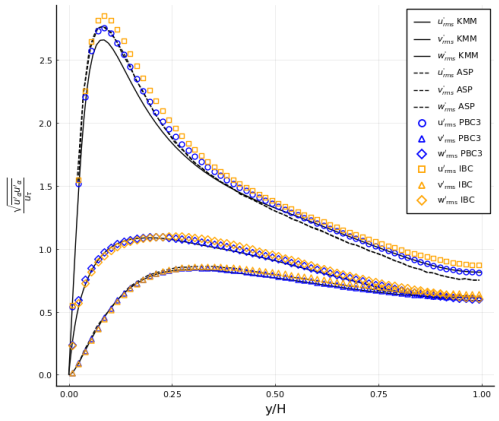
The previous section has shown that benchmark turbulence statistics are reliably obtained from the current GPU FM-LBM model. In this section the model is extended by replacing the periodic boundary conditions in the streamwise direction by more realistic inflow and outflow conditions. More specifically, the inflow turbulence profile is provided by running an auxiliary domain in tandem with the main simulation. To ensure reliable solutions are obtained, the auxiliary domain is chosen such that the numerical configuration of both domains is the same, except for the streamwise boundary conditions. In the auxiliary domain, a periodic boundary condition will be employed along the x-direction. During simulations, a plane slice of data from the auxiliary domain is directly used as an inflow condition in the main simulation. Lastly, a zero-gradient Neumann condition is prescribed at the outlet boundary of the main simulation to model the outflow of the turbulence. Before collecting statistics, both domains are simulated for at least one flow-through time to ensure the flow in the main domain is fully developed and consistent with inflow condition.

To validate the reliability of this new set of boundary conditions, the same low-order turbulence statistics as the previous section are retrieved from the main simulation domain. Based on the findings in section 4.3, the size of both the auxiliary domain and the main domain is chosen to be $512 \times 128 \times 128$. Furthermore, the grid resolution is set at $\Delta^+ = 2.8$. The auxiliary domain is therefore identical to simulation PBC3 from the previous section. Turbulence statistics are once again evaluated against the benchmark studies listed in Table 4.3, as well as to simulation PBC3.



(a) Mean streamwise velocity near the wall

(b) Reynolds stress profile



(c) RMS velocity fluctuations

Figure 4.4: Graphs showing turbulence statistics retrieved from simulations using a strong recycling method at streamwise location $x = 4H$. Figure (a) shows the mean streamwise wall velocity, (b) the Reynolds stress profile, and (c) the root mean square velocity fluctuations. Dotted dashed line corresponds to analytical velocity profiles in the viscous sublayer and the log layer.Furthermore, The solid and dashed lines represent results from Kim et al. [50] and Amati et al. [54] respectively. The colored data points are retrieved from the current model.

The results are presented in Figure 4.4. Here the analytical functions and the benchmark functions are once again depicted using dotted, dashed and solid lines. Furthermore, the data from PBC3 is displayed as the blue scatter points, while IBC represents the data from the realistic inflow model. Results from IBC are collected at streamwise location $x = 4H$, which equals half the domain length.

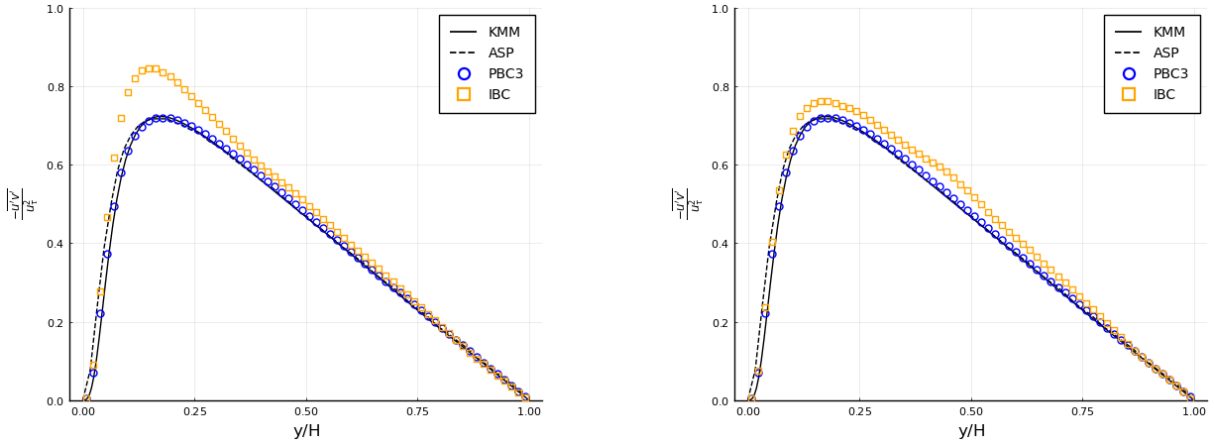
Mean streamwise velocity

Figure 4.4a shows the mean streamwise velocity in wall units, u^+ , as a function of wall-normal distance in wall units, y^+. It is clear the mean velocity profiles are near-identical and just slightly deviate from KMM in the near-wall region.

The Reynolds stress retrieved from the simulations is displayed in Figure 4.4b. The IBC curve follows the benchmark and PBC3 closely near the wall, but starts to display a slight deviation from the benchmark lines when starting towards the center of the channel. There is also a slight discrepancy observed for u'_{rms} in Figure 4.4c. Near its peak value an overshoot of approximately 3%.

Reynolds stress

The Reynolds stress retrieved from the simulations is displayed in Figure 4.4b. The IBC curve follows the benchmark and PBC3 closely near the wall, but starts to display a slight deviation from the benchmark lines when starting towards the center of the channel. There is also a slight discrepancy observed for u'_{rms} in Figure 4.4c. Near its peak value an overshoot of approximately 3% is observed.To investigate these findings further, the Reynold stress and RMS velocity fluctuations profiles for a position at the outlet and one near the outlet are displayed in Figure 4.5 and Figure 4.6a respectively.



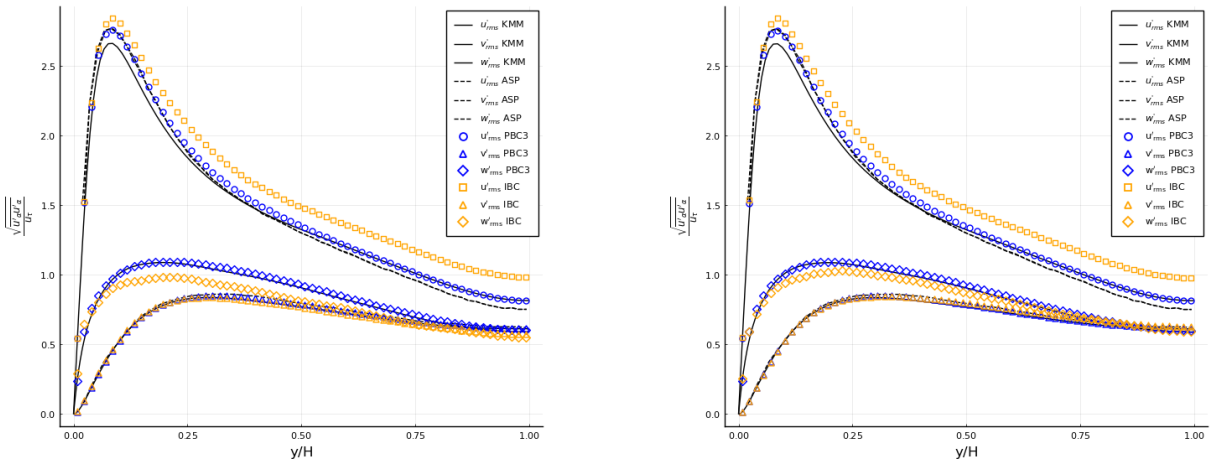
(a) Reynolds stress at outlet position $x = 8H$

(b) Reynolds stress at position $x = (63/64) \cdot 8H$

Figure 4.5: Graph showing the results for retrieving the Reynolds Stress profiles as function of distance to the wall. Figure (a) corresponds to a location at the outlet Figure (b) is a location a few grid points upstream. The solid and dashed lines represent results from Kim et al. [50] and Amati et al. [54] respectively.

From the Reynolds stress graphs it becomes clear that a deviation of approximately 14% is present. Figure 4.5b displays data that is positioned at few grid nodes upstream from the outlet. At this location, the peak value has moved towards the benchmark points, but now a slight increase of the Reynolds stress values is observed at a distance farther from the wall. Based on these graphs, it appears that the zero-gradient condition applied at the outlet reflects non-physical turbulence structures back into the domain. This problem is also experienced in other turbulent simulations employing outlet Neumann conditions [56]. A zero-gradient condition assumes that flow exits the domain with a smooth velocity field. However, in the buffer layer and log-layers strong local velocity gradients are present. In these regions, a zero-gradient does not reflect the natural evolution profile of the velocity. Judging from the overshoot of the peak Reynold-stress at the outlet, the simulation responds by creating artificial coherent turbulent structures. Because of the non-locality of turbulence flows, these new turbulent structures are transported upstream into the domain as proven by Cziesla et al. [56] and as observed from the discrepancy in Figure 4.5b.

RMS velocity fluctations The effect of the zero-gradient outlet condition on the RMS velocity fluctations at the outlet and close to the outlet is shown in Figure 4.6. At both locations, u'_{rms} is overestimated by the simulations. According to Cziesla et al. [56] entrainment of flow structures into the domain causes overshoots in RMS velocity fluctuations. This increase of turbulence intensity is further proof that non-physical turbulent structures are creaed by the Neumann outlet condition.



(a) RMS velocity fluctuations at outlet position $x = 8H$

(b) RMS velocity fluctuations at $x = (63/64) \cdot 8H$

Figure 4.6: Figure showing the results for RMS velocity fluctations value for a realistic inflow simulation. Figure (a) corresponds to curves located at the outlet Figure (b) shows curves a few points upstream. The solid and dashed lines represent results from Kim et al. [50] and Amati et al. [54] respectively.

4.5 GPU-PERFORMANCE

4.5.1 GPU performance indicator

The performance of the parallel programming on implementation is discussed in this section. To quantify the computational efficiency of a parallel LBM algorithm, a performance indicate called million lattice updates per second (MLUPS) is often used in literature [57]. The common definition is given by

$$\text{MLUPS} = \frac{N \times N_t}{T} \times 10^{-6} \tag{4.2}$$

Here, N denotes the number of grid points, N_t the total amount of LBM time steps simulated and T the simulation time expressed in seconds. The factor of 10^-6 is included so the value of MLUPS is given in mega LUPS.

A useful way to quantify the algorithms efficiency is by comparing the amount of MLUPS of an simulation to the maximum theoretical value that the GPU hardware can achieve. To this end, the maximum theoretical MLUPS of a GPU is defined by

$$\text{MLUPS}_{max,theory} = \frac{BW}{4N_{vars}} \times 10^{-6}. \tag{4.3}$$

Here, BW stands for the bandwidth of the GPU considered and N_{vars} is the amount of global memory accesses of a thread per time step [57]. In this work an NVIDIA GPU-A100 was used to perform the simulations.It is reported that its measured bandwidth amount to $BW = 1640\ GB$ [58]. In the flow simulation for LBM, global memory is accessed approximately 134 times in one time step. In theory, maximum bandwidth capacity is never reached in LBM applications. To create a more realistic efficiency parameter, it is recommended to use a percentage of roughly 80% of the GPU bandwidth specified [59]. Using this information, leads to $\text{MLUPS}_{max,theory} = 2485$

4.5.2 Simulation efficiency results

This section reports the achieved MLUPS values for the different simulations performed in this chapter. An overview of each simulation with its calculated amount of MLUPS is provided in Table 4.4. Additionally, a reference value from a similar study done by Spek [25] is added to the table.
From the table it is clear that a significant speed-up has been achieved compared to the study of Spek. For some simulations, the MLUPS count is almost three times as large. Furthermore, ratio of the MLUPS achieved to the maximum possible GPU bandwidth has also increased significantly. This indicates that the higher MLUPS count does not comes from the fact that a different GPU has been used compared to the reference study.

Table 4.4: Overview of the MLUPS achieved for each simulation performed in this chapter. A simulation of Spek [25] is included which is identical to PBC1.

Simulation	$N_{\text{Number of grid nodes}}$	MLUPS	GPU bandwidth used
PBC1	4.4e6	1022	41%
PBC2	1.3e7	1181	48%
PBC3	8.4e6	1048	42%
PBC4	2.5e7	1079	43%
IBC	1.7e7	1205	48%
Spek	4.4e6	403	19%

The speed-ups observed in the current GPU implementation over the DDF-FMLBM model presented in the reference study can be attributed to three key differences. First of all, the reference study was based in Pyton, while the current model is written in Julia. As discussed in Section 3.4, Julia offers improved computational performance compared to Python [48]. Moreover, Julia allows for the allocation of shared memory within GPU kernels. In the current GPU model, the solution vector in the FMLBM collision kernel is placed into shared memory, something which was not possible in the reference study. This significantly reduces the number of global memory accesses during the matrix multiplication, which is the most memory-intensive part of FMLBM model. Lastly, the current implementation includes an optimized way for iterating over either rows or columns of the filter matrix during the matrix multiplication step. This optimization strategy further improved the algorithm's performance.

Considering the fact that the effective GPU efficiency across simulations is about 45%, there is still room to improve the LBM algorithm. More efficient memory usage can for example be achieved when the collision and propagation kernels are combined, as shown in [60].

4.6 CONCLUSION

In this chapter, the DDF FMLBM was compared to the canonical case of DNS of turbulent channel flow. Turbulence statistics showed a good agreement with benchmark statistics. Furthermore, it was found that further refinement of the grid resolution lead to only a marginal increase in accuracy. Furthermore, choosing an appropriate domain length to allow the large scale eddies to evolve lead to improved turbulence statistics, especially in u'_{rms}.

Besides that it was found that the realistic inflow method produced realistic turbulent profiles. However, it was observed that the zero-gradient Neumann boundary conditions added non-physical at the outlet of the simulation.

The GPU performance has been measured via the number MLUPS. By switching to a Julia based code, placing the solution vector of the FMLBM method in shared memory and by optimizing the matrix multiplication strategy, significant speed-ups were achieved compared to the previous study by Spek [25].

5 VALIDATION OF FREEZING MODEL

This final chapter presents the results obtained from extending the FM-LBM model to include the simulation of phase change in thermal flows. To this end, a second distribution function is added to the model that solves the total enthalpy equation. The thermal model takes the turbulent flow fields as an input to calculate the effects of thermal convection on enthalpy fields. This chapter consists of two parts. In the first part, the thermal model is validated by running a benchmark case. The results are then compared to literature studies. The second part of this chapter will be the simulation of a spatially developing ice layer.
Section 5.1 outlines the relevant input parameters and the computational setup for the validation simulations. In addition, the benchmark studies are briefly reviewed. Following this, a canonical thermal flow case is used as a validation of the model alongside a phase change validation simulation in Section 5.2. After validation is performed, Section 5.3 presents the results of simulating spatially developing freezing in channel flow for different cold-plate temperatures. A conclusion of this chapter is given in Section 5.4

5.1 COMPUTATIONAL SETUP

In this chapter, a thermal channel flow setup is considered, which is depicted in figure 5.1. The domain length can be varied according to the scaling parameter α. The channel walls are located at $2H$ and the spanwise direction has a widht of $2H$. A constant temperature difference is imposed between the walls. The upper wall temperature is denoted by T_U and the lower wall temperature by T_L. A body force $\mathbf{f}$ is applied along the streamwise direction. Furthermore, buoyancy effects are not considered. This specifically means the flow field is independent from the thermal field. However, the thermal field needs input from the flow field to account for thermal convection. The FMLBM model from the previous section is extended by including a thermal distribution function in the DDF approach. The discrete velocity schemes used for both the flow field and thermal field is the D3Q19-scheme.

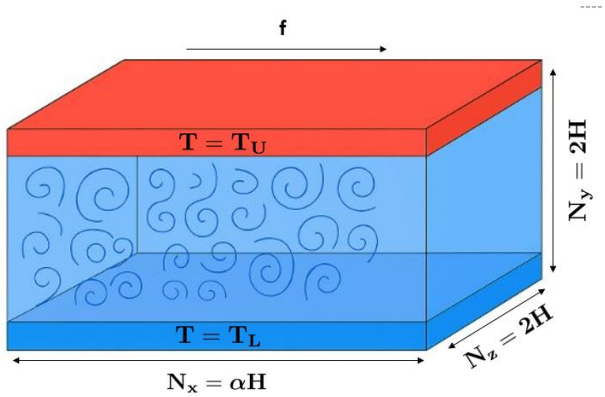


Figure 5.1: Schematic depiction of the channel being considered in the simulations in this section. A body force f is applied along the streamwise direction. α is a variable scaling parameter. Upper wall temperature T_U an lower wall temperature T_L are applied at the walls.

Thermal- and flow boundary conditions are considered periodic in the streamwise direction. At the walls a no-slip condition is imposed for the flow model, while a fixed temperature is set at the wall for the thermal model. This means that a Dirichlet condition needs to be enforced, which is achieved in a cell-centered LBM model via the anti-bounce back method. The imposed streamwise boundary conditions differ in this chapter. In Section 5.2 the streamwise direction is assumed to be periodic for both the flow- and thermal simulations. In Section 5.3 however, a spatially developing ice layer is investigated. To achieve this, the streamwise conditions for the flow field are the same ones used as in the realistic inflow simulations from the previous section. This means that the inflow uses a strong recycling method, while the outflow is modeled by a zero-gradient Neumann condition. The thermal inlet condition is an uniform temperature profile, implemented similarly as the thermal wall condition. The thermal outlet condition is taken to be a zero-gradient Neumann condition.

A complete overview of the flow and thermal input parameters is given in Table 5.1.The parameters are expressed in SI unit and LB units. Turbulent flow at Re_τ is studied alongside a Prandlt number of $Pr = 0.71$, which is the Prandtl number of air. This Prandtl number was chosen based on two reasons. The first one is that in a lot of benchmark studies, the Prandtl number of air is used as reliable reference value. The second reason is that in turbulent LBM simulations, a higher Prandtl number can quickly lead to numerical instability [25] [61]. Once again, properties of water at room temperature are implemented. This computational setup corresponds to the canonical benchmark cases [62] [63].

Table 5.1: Input parameters used in the simulations in this section. Flow and thermal properties are representative of water at room temperature. The freezing temperature, as well as the latent heat, are phase change properties at the freezing point of water.The Prandtl number corresponds to air.

Quantity	Symbol	Value (SI)	LB units
Friction Reynolds number	Re_τ	180	180
Prandtl number	Pr	0.71	0.71
Grid spacing	Δ^+	3.3×10^{-4} m	2.4
Density	ρ	$1000\,\mathrm{kg\,m^{-3}}$	1.0
Kinematic viscosity	ν	$1.7 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$	2.8×10^{-3}
Body force	f_x	$5.99 \times 10^{-3}\,\mathrm{m\,s^{-2}}$	5.8×10^{-7}
Specific heat (solid/liquid)	$C_{p,s/l}$	$2.1/4.2 \times 10^3\,\mathrm{J\,kg^{-1}\,K^{-1}}$	$6.2/12.5 \times 10^2$
Thermal diffusivity (solid/liquid)	$\alpha_{s/l}$	$9.58/2.39 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$	$15.9/4.0 \times 10^{-3}$
Latent heat	L	$3.34 \times 10^5\,\mathrm{J\,kg^{-1}}$	9.9×10^4
Freezing temperature	T_{freeze}	273.15 K	273.15

From the previous section, it is concluded that the grid resolution does not need further refinement. However, instabilities in the flow field were observed at $\Delta^+ = 2.8$ when temperature differences in the flow became too high. These instabilities did not lead to instable simulations, but are nevertheless undesirable.The smallest scale for thermal fluctuations is defined by the Batchelor scale from equation $\eta_B = \eta Pr^{-1/2}$ [62]. Assuming a Kolmogorov length scale of $\eta^+ = 2.0$, the minimal resolution to capture the thermal field at $Pr = 0.71$ is $\Delta^+ = 2.4$. In Figure A.1 in Appendix A two temperature snapshots taken at the same number of time steps are depicted. In the left graph, numerical instability is observed when a grid spacing of $\Delta^+ = 2.8$ is implemented. Refining the grid to $\Delta^+ = 2.4$ creates a stable field as can be seen from the right graph. In the remainder of this thesis, the grid spacing for both the flow field and thermal field will be set at $\Delta^+ = 2.4$.

5.2 VALIDATION OF THE FREEZING MODEL

5.2.1 Benchmarking of Thermal statistics

The accuracy of heat transfer by the DDF FMLBM thermal model using a D3Q19 scheme for both the momentum distribution and thermal distribution is discussed in this section. The channel flow setup from figure 5.1 is considered with $\alpha = 8$ and periodic boundary conditions in the streamwise- and spanwise directions for both the flow field and thermal field. Furthermore, a no-slip and a Dirichlet are applied at the walls for the flow field and thermal field, respectively. To retrieve steady-state thermal statistics, the temperature difference between the upper wall and lower wall ΔT is fixed. Upper wall temperature is set at $T_U = 280\ K$ and lower wall temperature at $T_L = 275\ K$. The initial temperature field of the fluid is uniformly chosen to be $T_0 = 277.5\ K$, corresponding to half the temperature difference. These temperature values are transformed into enthalpy values that serve as the input for the thermal distribution function. In order to obtain stable, physical thermal fields from the simulations, temperature and enthalpy values need to be scaled and stretched according to the transformation rules discussed in Section 3.2.3. Maximum and minimum sensible enthalpies are chosen to be 1 and 0 respectively in transformed units. Table 5.2 gives an overview of the necessary input values of the simulation expressed in SI units and in transformed units.

Table 5.2: Overview of thermal parameters and their transformed values used in the simulation.

Quantity	Symbol	Value (SI)	LB units	Transformed units
Upper wall temperature	T_U	280 K	280	8.03×10^{-4}
Lower wall temperature	T_L	275 K	275	0
Initial temperature	T_0	277.5 K	277.5	4.01×10^{-4}
Freezing temperature	T_{freeze}	273.15 K	273.15	-2.97×10^{-4}
Minimum sensible enthalpy	h_{min}	1.16×10^6	3.43×10^5	0
Maximum sensible enthalpy	h_{max}	1.18×10^6	3.49×10^5	1
Latent heat	L	3.34×10^5	9.91×10^4	15.19

The simulation in this study is first run for approximately 125 ETT to ensure a converged and developed thermal field. Statistics are obtained using an averaging time of $T_A = 56.3$ ETT and samples were taken every 1300 time steps. In total, the simulation needed $N_T = 2.1 \times 10^6$ time steps to finish. This corresponds to total simulation time of approximately 14 hours.

Two benchmark studies, which simulate the canonical case, are used to validate the thermal statistics. A DNS study on heat transfer in turbulent channel flow has been performed by Kawamura et al.[63] using a second-order finite difference method. In this study the grid resolution in the wall-normal direction Δy^+ was varied between very fine near the wall(0.40) to coarser near the centerline(11.5). The computational box that was used had dimensions $6.4H \times 2H \times 3.2H$. Based on these settings, it is assumed that this benchmark study is capable of capturing all the information within the flow, thus being a highly reliable benchmark. An MRT-LBM study by Ren et al. [62] uses a Large Eddy Simulation (LES) to model the turbulence behavior of the flow. This LBM study adopted a D3Q19 momentum scheme and a D3Q7 thermal scheme. A DDF-approach was used where the thermal distribution function retrieves flow temperature. A uniform grid spacing $\Delta^+ = 3.0$ is used throughout the domain. The domain size is not adequate enough to capture all turbulence scales, which is similar to the current model. All the specifics of the studies used, are summarized in Table 5.3.

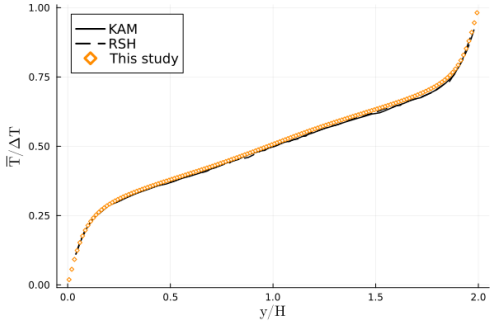
Table 5.3: Overview of simulations used to validate the DDF-FMLBM thermal model. KAM and RSH refer to the benchmark studies on heat transfer in channel flow from Kawamura et al[63] and Ren et [62], respectively. The amount of grid nodes in the y-direction in KAM is not known.

Simulation	Numerical technique	Collision model	$N_x \times N_y \times N_z$	Computational volume	Δy^+
KAM	2nd order FDM	–	$128 \times ? \times 128$	$6.4H \times 2H \times 3.2H$	0.40 – 11.5
RSH	LES-LBM	MRT	$256 \times 122 \times 96$	$4.3H \times 2H \times 1.6H$	3.0
DNS Thermal [This study]	LBM	FMLBM	$608 \times 152 \times 152$	$8H \times 2H \times 2H$	2.4

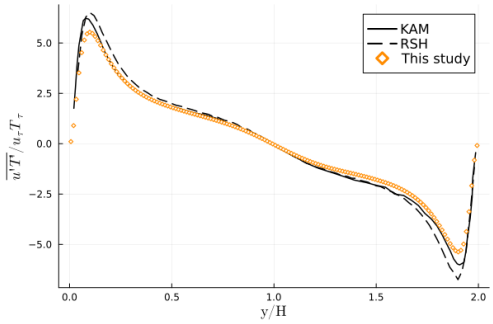
Thermal statistics of interest in this study are the mean temperature profile $\overline{T}$, streamwise turbulent heat flux $\overline{u'T'}$ and the temperature fluctuations $\sqrt{\overline{T'T'}}$. Figure 5.2 shows all these variables as a function of the wall-normal distance y normalized by the half-channel height H.

Mean Temperature

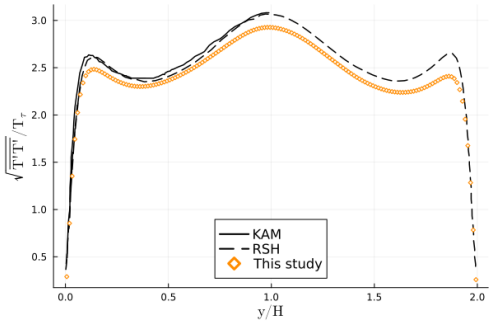
Figure 5.2a shows the mean temperature profile profile normalized by the temperature difference between the walls ΔT. From the graph, it is observed that the mean temperature profile accurately follows the curves of KAM and RSH. Near the walls, conductive heat transport is more dominate than convective heat transfer. Therefore, mixing through convection is limited and a steep temperature gradient is observed in this region. In the buffer layer and log layer eddies are created, which in turn transport the heat from the wall via convection. As a result of this transport, the slope of the curve becomes more smooth towards the center channel, which is readily observed from the graph.



(a) Normalized mean temperature



(b) Normalized streamwise turbulent heat flux



(c) Normalized RMS temperature fluctations

Figure 5.2: Graphs showing thermal statistics obtained from a benchmark simulation of the DDF-FMLBM model plotted as functions of distance to the wall. Figure (a) shows the normalized mean temperature profile scaled by $\Delta T = T_U - T_L$, (b) the normalized streamwise turbulent heat flux $\frac{\overline{u'T'}}{u_\tau T_\tau}$, and (c) the normalized root mean square temperature fluctuations.Solid and dashed lines represent reference data from Kawamura et al. [63] and Ren et al. [62] respectively. Orange markers display data from the current study.

Turbulent heat flux

The turbulent heat flux is a characteristic measure of the effect of turbulence on mixing. The results for this turbulence term are displayed in Figure 5.2b. As seen in the plot, the overall shape of the data points is similar to the benchmark lines. Turbulent mixing effects are strongest in the buffer layer, where anisotropic turbulence structures are created. At the center of the channel, turbulence becomes more isotropic meaning there is less directional preference of the fluid to transport the heat to. Therefore, the streamwise turbulent heat flux tends to zero towards the center of the channel. It is observed that the peaks at $x \approx 0.15H$ and $x \approx 1.85H$ are approximately 4% underestimated compared to KAM. The RSH line however shows a small overshoot at this peak, which rules out that the LBM technique is the cause of the undershoot. Further discussion of this deviation is conducted using the results from the RMS temperature fluctuations.

RMS temperature fluctuations

The orange scatter points in Figure 5.2c shows result corresponding to the RMS temperature fluctuations. The shape of the curve looks similar to both KAM and RSM. In the viscous sublayer near the walls, the temperature variance drops to zero as a result of reduced turbulent effects and the increased role of thermal diffusion in heat transfer. A total of three maxima can be observed in the plot. Two of them are situated in the buffer layers and are attributed to efficient turbulent mixing and steep thermal gradients in this region. The second and biggest maximum however, is present in the center of the channel and has a different cause. Fluid from near-wall is transported towards the center of the channel by turbulent eddies. Since both walls are fixed at different temperatures, a mix of hot of cold fluid is present in the core region. These thermal structures are then convected along the mean flow in the center channel, continually interacting and mixing with one another. As a result, high temperature variance is observed in the core region [64].

Furthermore RMS values of this study show a slight asymmetry around the centerline. This is noticed from the fact that the value at $y/H \approx 0.15$ is slightly bigger than the value observed at its symmetric location around $y/H \approx= 1.85$. This indicates that thermal field was not fully statistically stationary before collecting statistical data. It is not surprising that this asymmetry appears only in the temperature variance results, since the curve is more sensitive to variations in temperature than previously discussed statistics. Furthermore, the data points exhibit an undershoot across the whole height of the channel by 4% as compared to KAM. Interestingly enough, RSM curves shows a very well alignment with the KAM curve, again indicating that the LBM is reliable in capturing thermal behavior. Lluesma-Rodríguez et al.[65] mention that the minimal domain size needed to retrieve accurate low-order thermal statistics should be $2\pi H \times 2H \times \pi H$. The current width of the domain does not meet the requirement, leading to truncation of large turbulent structures. These structures play a key role in mixing, so truncating these can lead to a lower temperature variance across the domain. Finally, reduced temperature fluctuations inherently lead to lower correlation values between velocity components, which will be especially apparent in regions with a high correlation. This is reflected in the streamwise turbulent heat flux shown in Figure 5.2b, where an under-prediction is observed near maxima.

5.2.2 Analytical expression for steady-state freezing

An analytical expression for a final ice layer thickness under convection influences can be derived by considering the a steady-state freezing scenario in a channel depicted in Figure 5.3. A similar reasoning will be followed to derive the expression as in the thesis of Spek [25]. In this picture a thermal fluid flows is present between flat plates. Two distinct regions can be identified, namely the pink region where the fluid is in a liquid state and the blue region where the fluid has turned solid. Furthermore, the channels height is represented by parameter L and the thicknness of the ice by d. In this particular case the walls are kept at constant temperatures, the upper wall at T_u and the lower wall at T_L. Also the walls are separated by a distance L and an ice layer has developed with thickness d. Furthermore, T_i and T_B are the interface- and bulk flow temperature, respectively.

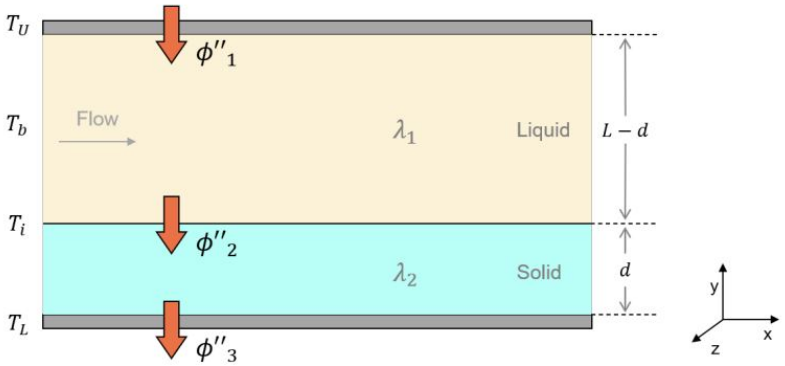


Figure 5.3: Steady state freezing within channel flow. [25]

A constant heat-flux is imposed in the y-direction as a result of a fixed temperature difference between the wall and by symmetry of the channel. This is captured in the statement

$$\phi_1'' = \phi_2'' = \phi_3'', \tag{5.1}$$

where ϕ_1, ϕ_2 and ϕ_3 correspond to the heat fluxes at the upper wall, the interface of the ice and the lower wall respectively. Fourier's law relates the heat flux to the temperature gradient [32]

$$\phi'' = -\lambda \frac{\partial T}{\partial y}, \tag{5.2}$$

where λ represents the heat conductivity in the material. In the ice, no fluid is moving so heat transfer is solely determined by conduction. Thus, the heat flux out of the lower wall can be defined according to Newton's cooling law [32]

$$\phi_3'' = h_3 A(T_i - T_L) \tag{5.3}$$

where $h = \frac{\lambda}{d}$ is the heat transfer coefficient. One of the main characteristics of turbulent flow is its effectiveness in thermal mixing. As a result, the temperature profile between the interface of the ice and the upper wall is constant at bulk temperature T_b. Heat fluxes ϕ_1 and ϕ_2 are solely determined by the temperature difference with respect to the bulk flow

$$\phi_1'' = h_1(T_U - T_b), \quad \phi_2'' = h_2(T_b - T_U) \tag{5.4}$$

where h_1 and h_2 are once again heat transfer coefficient. An universal expression for h is given by [32]

$$h = \frac{\lambda}{L} Nu. \tag{5.5}$$

Here L symbolizes the characteristic length and Nu is Nusselt number, which is a non-dimensional number that indicates the relative importance of convection on the heat transfer within a system. It is defined as the ratio between convective heat transfer and the total heat transfer. When only conduction is considered, the Nusselt number is equal to 1. Consequently, the relevant heat transfer coefficients can be defined by

$$h_1 = \frac{\lambda_1}{L-d} Nu_1, \quad h_2 = \frac{\lambda_2}{L-d} Nu_2, \quad h_3 = \frac{\lambda_3}{d} \tag{5.6}$$

where Nu_1 and Nu_2 correspond to the Nusselt numbers of ϕ_1'' and ϕ_2''. Now, following the mathematical procedure outlined by Spek [25], the expression for the steady-state ice thickness is obtained:

$$d = \frac{A}{1+A} L \tag{5.7}$$

Here A is a non-dimensional geometric parameter which reflects the effects of convection, material properties and the temperature differences on the steady-state ice thickness. When constant density is assumed, parameter A is expressed by

$$A = \left(\frac{1}{Nu_1} + \frac{1}{Nu_2}\right) \frac{C_{p,1}\alpha_1}{C_{p,2}\alpha_2} \frac{T_i - T_L}{T_U - T_i}. \tag{5.8}$$

Now, the combination of Eq. 5.7 and 5.8 enables the calculation of an analytical value for the ice thickness in channel flow with a constant temperature difference across the channel.

5.2.3 Validation of Phase Change Implementation

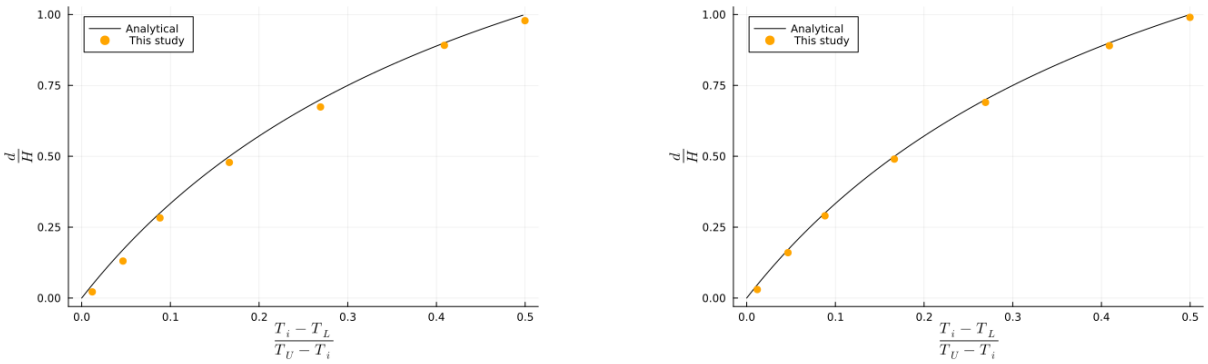
To asses the ability of the DDF FMLBM model to accurately simulate solidification phenomena, a steady-state freezing simulation with zero velocity is performed. The physical setup corresponds to the analytical steady-state freezing problem described in section 5.2.2 for channel flow setup similar to figure 5.1. For the analytical expression, equations 5.7 and 5.8 are used to calculate the analytical ice thickness. The characteristic length of the channel is $L = 2H$. Nusselt numbers are equal to unity when no convection is present, reducing the expression for geometric parameter to $A = \frac{C_{p,s}\alpha_s}{C_{p,l}\alpha_l} \frac{T_i - T_L}{T_U - T_i}$. During the simulations, the liquid fraction is used to track the local phase state of the fluid. As solidification set in, latent heat associated with melting is numerically subtracted from the local total enthalpy. Furthermore, a no-slip condition is enforced for locations within ice ($f_l = 0$) by employing the immersed boundary method. Since there is no turbulent velocity field initialized in the channel, and conduction is effectively one-dimensional in the y-direction due symmetry, the grid size is chosen to be $2 \times 96 \times 2$ to reduce computational cost. Input thermal- and flow properties are taken from Table 5.1.

Table 5.4: Different lower temperatures applied in freezing simulations. Upper wall value is fixed at $T_U = 300\ K$ and the interface temperature corresponds to that of freezing water $T_i = 273.15\ K$. $(T_i - T_L)/(T_U - T_i)$ represents the temperature fraction. Transformed temperature values are also shown, using $\tilde{h}_{\min} = 0$ and $\tilde{h}_{\max} = 1$ as the limiting enthalpy values.

Simulation no.	T_L (K)	$\frac{T_i - T_L}{T_U - T_i}$	$\tilde{T}_L$ (IK)	$\tilde{T}_i$ (10^{-6} IK)	$\tilde{T}_U$ (10^{-6} IK)
1	272.83	0.0118	0	2.82	240
2	271.90	0.0464	0	10.8	244
3	270.78	0.0882	0	20.1	248
4	268.68	0.167	0	36.6	256
5	265.92	0.269	0	56.5	266
6	262.17	0.409	0	80.8	279
7	259.73	0.500	0	95.2	286

In this validation simulation, the temperature of the cold wall is varied to test the effects of different freezing intensities. An overview of the lower wall temperatures used in this study, alongside the corresponding temperature fractions, is given in Table 5.4. The top wall temperature and the bulk temperature are set at $T_u = 300\ K$ and $T_0 = 285\ K$, respectively. The temperature at the interface is $T_i = 273.15\ K$, which is the freezing temperature of water. Corresponding transformed temperature values are also reported in the table.

Figure 5.4a shows the result of the freezing simulation using 96 grid points in the wall normal direction. The plot shows the steady-state ice thickness normalized by the channels half height as a function of the temperature fraction. From the graph, deviations from the analytical expression on the order 1 grid point are discerned. All the data points are below the analytical line, a similar discrepancy was also observed in the DDF FMLBM model by Spek [25]. To check whether this discrepancy is an inherent problem to the DDF model or just a resolution issue, an additional simulation is run using $N_y = 200$ grid points in the y-direction. From Figure 5.4b it is observed that the scatter points have moved significantly closer to the analytical line, but still a slight error is discerned. This indicates that by choosing a finer grid, the current model is able to reproduce the analytic steady-state ice thickness for a system relying solely on diffusive heat transfer.



(a) Zero-velocity with $N_y = 96$

(b) Zero-velocity with $N_y = 200$

Figure 5.4: Graphs showing results for a zero-velocity steady-state freezing simulation for different cold-plate temperatures, expressed on the x-axis by a temperature fraction $\frac{T_{Freeze} - T_L}{T_U - T_{freeze}}$. On the y-axis the steady-state ice thickness normalized by the channels height is displayed.Figure (a) shows the simulation using $N_y = 96$ and (b) show the simulation using $N_y = 200$.

5.3 SIMULATIONS OF A SPATIALLY DEVELOPING ICE LAYER

This section presents the results of simulations of a spatially developing ice layer in a turbulent channel flow. To this end, two simulation domains are considered. The main simulation domain corresponds to the one where the ice layer is developing. In this domain, both the flow field and the thermal field are solved for using the DDF-FMLBM model. The situation in the main domain corresponds to the one depicted in figure 5.1 using $\alpha = 4$. Boundary conditions in the wall-normal and spanwise directions are the same as in the simulations of the previous section. The inflow condition for the flow field is provided via the strong recycling method using flow information from an auxiliary domain. An equithermal temperature profile is imposed at the inlet for the thermal field. Furthermore, both the flow and thermal outlets are modeled using a zero gradient Neumann condition. Lastly, the immersed boundary method is used to impose a no-slip condition in the ice.

The auxiliary domain is setup as depicted in figure 3.4 with $\alpha = 8$. In this domain, only the flow field is solved. Periodic boundary conditions are imposed in the streamwise- and spanwise directions and a no-slip in the wall-normal direction . The length of the main simulation domain is chosen to be twice as small as the auxiliary simulation, in order to simulate more time steps. Shortening the domain length is justified, as realistic inflow and outflow conditions allow for the natural evolution of the streamwise eddies.

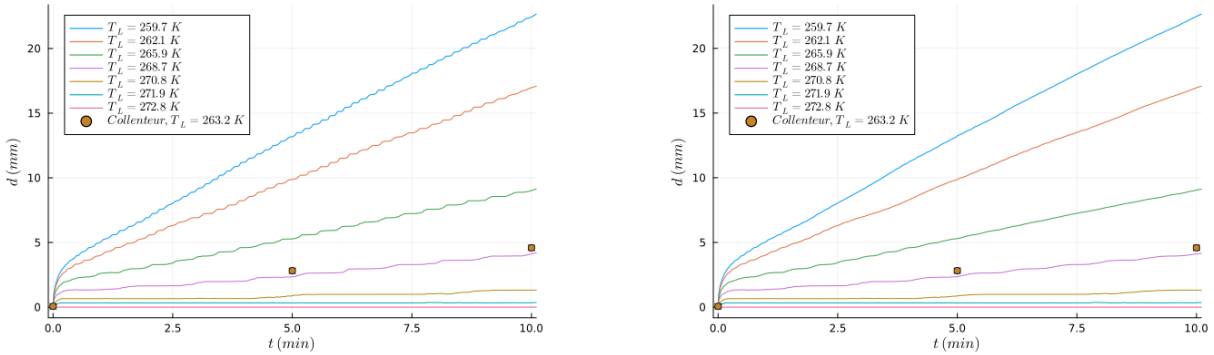
Within the main simulation domain, a fixed wall temperature difference is applied. Multiple simulations are performed with varying cold wall temperatures to investigate the effect on the developing ice layer. The same cold wall temperature values are used as in the zero-velocity freezing experiment, these are listed in Table 5.4. Additionally, the input parameters from Table 5.1 are used for the thermal and flow properties. This means that the fluid being simulated has the properties of water at room temperature, but the Prandtl number of air. The limitation in simulating higher Prandtl numbers, such as that of water, is due to inherent instability issues in the thermal FMLBM. The simulations are run for 25 hours for a total of 3.5 million time steps. This amounts to a total of 10 minutes of total time.

Figure 5.5 shows the spanwise-averaged ice thickness sampled at mid-length location $x = 2H$ as a function of time. The legend indicates which curve corresponds to each cold wall temperature. For comparison, experimental data from the thesis of Collenteur [30] is included in Figure 5.5a represented by the orange markers. In that study, ice layer growth in channel flow was experimentally investigated using different Reynolds numbers and cold wall temperatures.

The experimental setup differs from the present numerical configuration in several important ways. For both studies, relevant numerical input parameters and the dimensions of the domains are summarized in Table 5.5. In the study of Collenteur, the upper wall is considered adiabatic, taking on the temperature values of the near-wall bulk flow. Further caution should be taken when comparing the results, since the turbulent velocity field was reported to be not sufficiently developed. Due to all the discrepancies in configuration and flow properties, only qualitative comparisons should be made between the two studies.

Table 5.5: Comparison between numerical parameters adopted in this study and those reported in the experimental study performed by Collenteur [30].

Quantity	Symbol	This study	Experimental study
Domain size	$N_x \times N_y \times N_z$	$0.10 \times 0.05 \times 0.05$ m	$1.45 \times 0.05 \times 0.05$ m
Reynolds number	Re	5530	6181
Prandtl number	Pr	0.71	13.4
Upper-wall temperature	T_U	300 K	*Adiabatic*
Lower-wall temperature	T_L	262.2 K	263.15 K
Inlet temperature	T_{in}	285 K	276 K
Kinematic viscosity	ν	$1.79 \times 10^{-6}\,\text{m}\,\text{s}^{-2}$	$1.70 \times 10^{-6}\,\text{m}\,\text{s}^{-2}$
Specific heat (solid/liquid)	c_p	$2.10/4.20 \times 10^{3}\,\text{J}\,\text{kg}^{-1}\,\text{K}^{-1}$	$2.10/4.22 \times 10^{3}\,\text{J}\,\text{kg}^{-1}\,\text{K}^{-1}$
Thermal diffusivity (solid/liquid)	α	$9.58/2.39 \times 10^{-6}\,\text{m}^2\,\text{s}^{-1}$	$11.2/1.34 \times 10^{-7}\,\text{m}^2\,\text{s}^{-1}$
Latent heat	L	$3.34 \times 10^{5}\,\text{J}\,\text{kg}^{-1}$	$3.33 \times 10^{5}\,\text{J}\,\text{kg}^{-1}$
Velocity profile	$\mathbf{u}$	Fully developed	Not fully developed
Centerline velocity	u_c	$0.22\,\text{m}\,\text{s}^{-1}$	$0.28\,\text{m}\,\text{s}^{-1}$
Sampling location	x_s	0.05 m	0.75 m



(a) All cold wall temperatures

(b) All cold wall temperatures, smoothed ice thickness

Figure 5.5: Graphs displaying spanwise averaged ice thickness in millimeters sampled at half the channel's length $L_x = 2H$ as function of freezing time in minutes. Figure (a) shows data from all temperature simulations including results from Collenteur [30] depicted as marker points, figure (b) shows the ice thicknesses averaged over 17 grid points in the streamwise direction. The standard deviation of each measurement point of the data from Collenteur is $\pm 0.2mm$ and is depicted with error bars.

Observing the curve in Figure 5.5a, ice grows rapidly in a matter of seconds. This rapid initial growth is attributed to the fact that no ice has formed yet. An ice layer acts as a thermal resistance, which inhibits heat extraction from the fluid. When the ice layer begins to form, the freezing process slows down. After this initial startup phase, the slope decreases and becomes somewhat constant for the cold-plate temperatures ranging from $T_L = 259.7\ K$ to $T_L = 268.68$ till the end of the simulation. The higher lower wall temperatures $T_L = 272.83\ K$ and $T_L = 271.90\ K$ flatten out, indicating that a steady-state situation is reached. The other curves however, still show transient behavior judging by the increasing ice thickness. Furthermore, curves display a wiggling behavior when entering the constant growth phase. These oscillating effects most likely stem from local turbulence mixing effects. This observation is further corroborated by performing a slight averaging in the x-directions over an interval of 17 grid points to smooth the effects of local turbulent fluctuations. Figure 5.5b shows that the oscillations are reduced, indicating turbulent heat transport is responsible for the oscillating curves. Furthermore, a colder lower wall temperature leads to faster ice growth as is expected.

In the figure, the marker points represent measurements of the experimental study. The error on each measurement point is $\pm 0.2\ mm$, which is included in the plot with error bars [30]. As can be seen from the graphs, measurements took place every 5 minutes, leading to a total of 3 data points between 0 to 10 minutes. As discussed, the modeled curves show a steep increase in ice growth in the first few seconds of the freezing simulations. It is hard to directly determine whether the experimental trend at the start of the freezing experiment is similar to that of the model, since there are not enough data points to capture the behavior in this time interval. Considering data point two and three, a linear trend can be observed especially when also taking into account data points four and five from the work of Collenteur [30]. When drawing a line between the two points and extrapolating this line backwards, it is observed that this extrapolated line crosses the d-axis somewhere for $d > 0\ mm$. This indicates that also in the freezing experiment there is a small interval where a steep ice growth gradient is present. After this initial interval, a linear trend is observed in the experimental data. Therefore, the behavior of freezing in the model is qualitatively similar to the experimental data. Also, no steady-state ice thickness is observed within 10 minutes of the experiment. Because of this, no conclusions can be drawn on whether the model correctly predicts the final ice thickness. To validate the steady-state ice thickness with the experimental data under consideration, roughly 180 minutes of freezing time must be simulated based on the final ice thickness achieved in Collenteur[30].

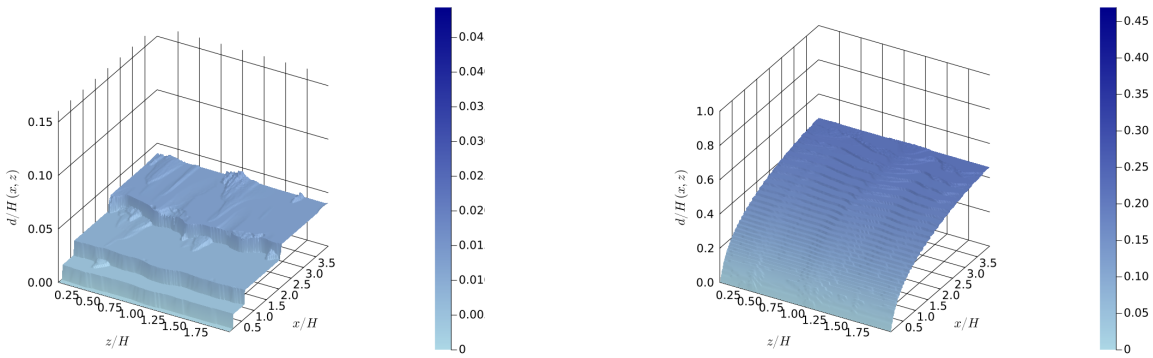
The curve that matches the temperature value of the experimental data best, is the orange curve with $T_L = 262.17\ K$ which is one degree colder. From Figure 5.5a it can be observed that the orange data points display a growth rate of around $0.32\ mm/s$, while the modeled curve grows at roughly $1.4\ mm/s$. This indicates that the simulation predicts an ice growth rate roughly four and a half times higher than the one observed experimentally.

To determine whether or not this increased ice growth rate is realistic considering the discrepancies in the setup of the simulation and the freezing experiment, is discussed now. The temperature setup of the experimental study would indicate that freezing should be stronger than the modeled case. The initial bulk temperature is 10 degrees celsius lower than the one used in simulations, which means that ice is more readily formed. Additionally, the upper wall temperature of the numerical simulation is fixed at 300 K continuously supplying heat to the fluid, further inhibiting ice growth. Another observation is that the sampling location in the study of Collenteur was taken 7.5 times further downstream than this study. As a result, the sampling location of this study is much closer to the warm inlet, which results in less ice growth as when a similar sampling location as the experimental study was chosen. Based on these observations, the actual ice growth rate should be bigger in the experimental case.

When considering the effects of the initial flow profile on ice growth, two observations can be made. First of all, the fluid's velocity is faster in the experiment as compared to the model. A higher velocity means that the fluid spends less time near the ice interface, de-accelerating ice growth in the experiment as compared to the model. Furthermore, it was reported that the velocity profile was not yet fully developed at the sampling location in the study of Collenteur. Therefore it is assumed that in the the model, where a fully developed profile was present, extra velocity and heat is provided to the region near the ice's interface, as compared to the experiment. From this, the ice growth in the model is inhibited.

Lastly, the Prandtl number used in the model is significantly lower than in the experimental study by Collenteur. For a fixed kinematic viscosity, this implies that the model exhibits a higher thermal diffusivity in both the solid and liquid phases compared to the experimental case. The primary means by which ice layer grows is through heat conduction. Thermal conductivity is defined as $\lambda = \alpha \rho C_p$, thus at the same density and specific heat, an increase in thermal diffusivity leads to an increase in ice growth rate of the model compared to the experiment. Still, it is difficult to draw a definitive conclusion, primarily because neither the extent to which the higher thermal diffusivity affects the simulation nor the effects of discrepancies in the experimental setup can be quantitatively determined given the current information. It is therefore recommended to perform numerical simulations that precisely models the experimental setup to better compare the model to experimental findings. This involves for example finding a way to simulate higher Prandtl numbers and implementation of an adiabatic boundary condition applied to the top wall.

To assess the influence of turbulence on the shape of the ice layer, two three-dimensional plots are shown in figure 5.6. Here ice thickness, normalized by the channel's half-height, is displayed as function of the spatial coordinates x and z. The left corresponds to a snapshot from the simulation using $T_L = 270.8\ K$, while the right plot shows data from simulation with $T_L = 262.1\ K$. Both snapshots are taken at $t = 5\ min$. Both ice layers show a spatially developing freezing front, whereas the ice layer is much ticker for the curve using a lower wall temperature. Furthermore, it is evident that the graph with the lower wall temperature shows a more uniform freezing interface. This suggest that at $T_L = 262.1\ K$, the solidification process is primarily driven by conduction. At higher wall temperatures, however, the ice front exhibits more irregularities, which are likely caused by variations in local heat transfer due to turbulent fluctuations.



(a) 3D ice thickness for $T_L = 270.8\ K$

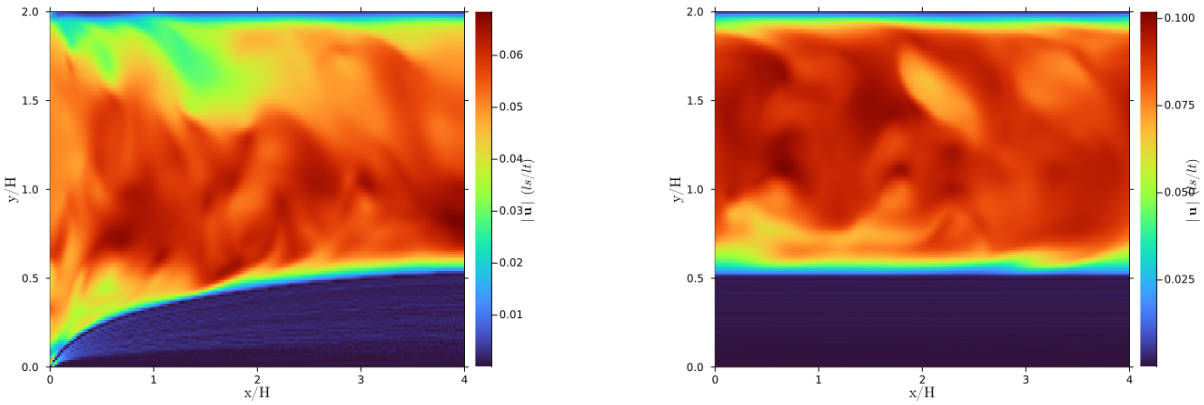
(b) 3D ice thickness for $T_L = 262.1\ K$

Figure 5.6: 3D surface plots of the layer thickness as a function of x and z normalized by the half channels height. Figure (a) shows an ice layer corresponding to $T_L = 270.8\ K$ and (b) displays an ice layer for a simulation with $T_L = 262.1\ K$. Snapshots are taken at t = 5 minutes.

A peculiar effect is observed in Figure 5.7, where the magnitude of the velocity is displayed in a two-dimensional plot in the x- and y direction. The left graph shows a velocity profile snapshot taken from the simulation with $T_L = 262.1\ K$. As can be seen, the fluid's velocity within the ice is non-zero. This is a non-physical result, implying that the no-slip condition is not properly enforced. The right graph in the figure shows a velocity snapshot of the same simulation, but here periodic boundary conditions are applied. The snapshots are taken at the same amount of time steps. Interestingly, the velocity within the ice is zero in the periodic simulations. Besides that, it is observed that for all simulations with realistic inflow conditions, the velocity within the ice becomes non-zero at some point during runtime. From this it can be concluded that either the turbulent inflow condition or the zero-gradient Neumann condition influences the velocity within the ice. Based on observations of the two-dimensional velocity profiles across all wall temperatures using realistic inflow conditions, it is evident that the velocity in the ice becomes largest near the bottom-left of the domain. This observation strongly suggests that the non-zero velocity in the ice is caused by the realistic inflow condition. During the simulation, the turbulent inflow, provided by an auxiliary simulation, does not feel the presence of the solid nodes developing in the main domain. Since these solid nodes do not exist in the auxiliary domain, the velocity field imposed at the inlet of the main domain remains unaffected by the growing ice layer. Therefore at the inlet, the turbulent velocity profile is not entirely synchronized with the no-slip condition that the immersed boundary method enforces in the main domain. This discrepancy is particularly pronounced at the first few solid nodes near the inlet at the wall, where the imposed turbulent profile introduces more momentum than the IBM expects and is able to suppress. As a result, velocity residuals build-up at the first solid nodes in the domain which are streamed into the ice. Furthermore, it is observed that the velocity at the left bottom corner of the domain grows over time, indicating that the IBM becomes less effective at imposing the no-slip condition here in the presence of residual build-up. As a result, non-zero velocity also becomes more present over time in the ice. All-in all, IBM is limited in its ability to enforce the no-slip condition in the presence of an inflow condition that is not corrected for the effect of the developing ice layer. To resolve this issue, the inflow profile should be imposed in such a way that it takes into account this evolving ice layer.

Based on the discrepancies in the turbulence statistics found near the outlet in Section 4.4, the Neumann outflow condition could also potentially be a source of error. It is therefore recommended to extend this research to include outlet conditions that are more suitable for handling the wide varieties of eddies leaving the fluid domain. For example, convective boundary conditions [26] could be used to model the outlet.

Furthermore, it should be noted that a non-zero velocity in the ice enhances heat transfer allowing more heat flux through the ice. As a result, the ice growth rate in the simulation is increased.



(a) Realistic inflow domain

(b) Periodic domain

Figure 5.7: Two-dimensional plots of the velocity magnitude plotted along the x- and y- direction. Figure (a) shows the velocity magnitude for a realistic turbulent inflow scenario [30] at $T_L = 262.1\ K$ and (b) shows the velocity magnitude for a periodic simulation at $T_L = 262.1\ K$. Snapshot are taken at the same number of time steps.

5.4 CONCLUSION

In this chapter, it has been shown that the DDF FMLBM model developed in this thesis is able resolve key features of turbulent heat transport and phase change behavior of water. Furthermore, this validated model was used to simulate spatially developing ice layers for different cold wall temperatures over time. From the model, ice thickness as function of time was retrieved. The physical total time simulated corresponded to 10 minutes. Most ice layers did not show a convergence to a steady-state during this 10 minutes. The results could only be compared to a single reference curve. The overall freezing trend between the experimental curve and the corresponding showed similarity. However, the simulation predicts faster freezing than its experimental counterpart. This discrepancy could not be meaningfully quantified, as many of the experimental conditions did not match those of the simulation. It is therefore recommended to extend the model to run a numerical simulation that mimics the experimental one.

Furthermore, it was observed that a residual velocity was present in the ice layer. Since the IBM works perfectly fine in periodic simulation, the realistic inflow profile is suspected to introduce velocity artifacts in the ice. It is therefore recommended to implement a mechanism so the inflow profile takes the effects of the developing ice layer on flow field evolution into account.

6 CONCLUSIONS & RECOMMENDATIONS

In this thesis an a GPU-accelerated DDF FMLBM model has been created, that is able to simulate key characteristics of turbulent flow and heat transfer phenomena using a direct numerical simulation approach. Furthermore, realistic streamwise boundary conditions have been implemented with the goal of modeling a spatially developing ice layer under turbulent flow conditions. This final chapter will present the conclusions drawn from this study and will provide recommendations regarding future research.

6.1 EFFECTS OF GRID RESOLUTION AND DOMAIN LENGTH ON TURBULENT STATISTICS

In this research, a direct numerical simulation (DNS) approach was applied to solve turbulent channel flow at $Re_\tau = 180$ using periodic streamwise boundary conditions. To achieve a fully resolved DNS, the grid resolution of the simulation should be fine enough to capture the effects of the smallest scales in the flow, while the grid size should be large enough to accommodate the largest turbulent structures. Guidelines on the resolution and minimal channel required to accurately reproduce turbulent flow effects are given by Moin et al. [13] and Flores et al. [52]. This study was however restricted to just two datasets that could provide fully developed initial velocity fields. The effect of grid refinement down to the Kolmogorov scale was investigated. Furthermore, the effect of the extent of the domain length on the accuracy of the turbulence simulations was also investigated. It was found that refining the grid size from $\Delta^+ = 2.8$ to $\Delta^+ = 2.0$ did not yield significantly better results, especially considering the increased computational costs. Furthermore, it was found that increasing the extent of the domain in the streamwise direction from $L_x = 4H$ to $L_x = 8H$, yielded visibly better results when compared to the benchmark statistics. This conclusion was in line with the minimal channel requirements set by Flores et al. [52]. Unfortunately, the spanwise extent of the domain was still too narrow for the simulation to be considered a fully resolved DNS.

6.2 IMPLEMENTATION OF REALISTIC STREAMWISE BOUNDARY CONDITIONS

To simulate a spatially a developing ice layer, realistic streamwise boundary conditions are required instead of periodic boundary conditions. It is important that the inflow condition contains all the relevant turbulent scales in order for flow field to stay developed. To this end the temporal strong recycling method was used, in which an auxiliary periodic simulation is run in tandem with the main simulation. A plane slice of the flow field was taken from the auxiliary domain as an inlet condition to the main simulation. Furthermore, the streamwise outlet condition was modeled using a zero-gradient Neumann condition. For this model, the conclusions on the grid resolution and spatial extent from the previous section were applied. While turbulence statistics agreed well with benchmark data away from the outlet, notable deviations were observed at and near the outlet, characterized by an overshoot in both the peak Reynolds shear stress and the streamwise velocity fluctuations, which also affected nearby upstream regions. The location of the reported overshoot coincided with the buffer layer and inner logarithmic sublayer, which are regions that contain a lot of small-scale velocity fluctuations due to high turbulence intensity. A zero-gradient does not model these fluctuations correctly, resulting in the formation of non-physical eddies that propagate back into the domain, affecting the simulations accuracy.

6.3 ACHIEVED COMPUTATIONAL SPEED-UPS

The GPU performance has been measured via the number of MLUPS. By switching to a Julia based code, placing the solution vector of the FMLBM method in shared memory and by optimizing the matrix multiplication strategy, significant speed-ups were achieved compared to the previous study by Spek [25].

6.4 VALIDATION OF FREEZING MODEL

In this chapter, the FMLBM model was extended to include a thermal distribution function employing a total enthalpy formulation. This DDF FMLBM was then validated against a DNS benchmark study. It was reported that heat transfer characteristics were reproduced with satisfactory accuracy, aside from a 4% under prediction in temperature variance and a minor discrepancy in the maximum streamwise turbulent heat flux. This undershoot is attributed to the inadequate spanwise extent of the simulation domain. Furthermore, it was noticed that for thermal simulations with a grid size of $\Delta^+ = 2.8$ temporary thermal instabilities could arise, when initial temperature gradients were set too high. Refining the grid to $\Delta^+ = 2.4$ resolved the issue. The final validation involved simulating a steady-state freezing under zero-velocity condition using the Immersed Boundary Method (IBM). The ice layer thickness was tracked by updating the liquid fraction on each lattice node. When the wall-normal grid resolution was sufficiently fine to make the relative impact of the inherent one-grid-node error in the liquid fraction variable negligible, the analytical steady-state ice thickness was accurately reproduced, indicating the effectiveness of the IBM.

6.5 SPATIALLY DEVELOPING ICE LAYER MODEL

The main goal of this thesis was to develop a model that can simulate freezing under realistic streamwise boundary conditions for varying cold wall temperatures under turbulent flow conditions. To this end, seven different lower wall temperatures were selected to investigate the effects on the ice layer thickness and shape. In the end, the simulation was run for 25 hours totaling 10 minutes of physical freezing time. The numerical results were quantitatively compared to a single experimental realization from Collenteur [30]. On overall, the global trends seen in the ice layer thickness growth over time was similar to the experimental results. However, it was reported that the ice growth rate of the model at an almost identical Reynolds number and cold wall temperature was four and a half times as fast as expected based on the reference study. Based on observations made in this study, it is likely that this discrepancy arises due to failure of the model to capture physically sound freezing effects rather than this discrepancy arising from the fact that an unrealistically low Prandtl number was used. Definite conclusions are nevertheless hard to make, since no similar numerical experiment has been performed. It was further reported that the ice shape fronts corresponding to higher cold wall temperatures showed an increased presence of irregularities on the developing ice interface as compared to lower cold wall temperatures. Furthermore, it was observed that a residual velocity was present in the ice layer. Since the IBM works perfectly fine in periodic simulation, the realistic inflow condition is suspected to introduce velocity artifacts in the ice, due to its inability to capture the effects of the developing ice layer on the flow field. It is therefore recommended to either implement a correction of the inflow profile in the main domain, or add the effect of the developing ice layer in the auxiliary domain.

6.6 RECOMMENDATIONS

Based on the findings in this theses, suggestions are given for future research. These recommendations can be grouped in turbulence modeling, boundary condition treatment, GPU implementation and realistic thermal modeling.

Turbulence Modeling

1. As discussed in the introduction, Reynolds numbers within the heat exchanger can reach Reynolds numbers up to the order of 10^5 [12]. In this thesis, only a fairly low turbulent Reynolds number of 5530 was investigated. Therefore it would be interesting to extent this model to be able to simulate higher Reynolds numbers. In this way, more data from Collenteur [30] can be used to validate the model. However, computational costs grow fast if one want to solve turbulent flows using DNS, due to the need of grid refinement to capture the effects of the Kolmogorov scales. An alternative way to model turbulence is by making use of a Large Eddy Simulation (LES), which resolves only the large scales within the turbulent flow, while modeling the effects of the small scales [31]. In this way, the the resolution requirement at higher Reynolds numbers is relaxed.

GPU-Implementation

1. In this study a significant speed-up was reported compared to previous studies. The average parallel efficiency rate was reported to be around %45 for the flow simulations. This means there is still room for improving the GPU-implementation. For example, it is possible to combine the collision and propagation kernels as to reduce the amount of memory overhead [60].

Boundary Conditions

1. It is recommended to implement more suitable outflow boundary conditions that allow for the natural exit of turbulence through the outlet. The zero-gradient Neumann condition does not facilitate this. One example of a potentially suitable boundary condition is the convective boundary condition, that retrieved accurate results in the FMLBM laminar freezing study of Bus [26] .

2. To accurately enforce the no-slip condition within the ice layer using the immersed boundary method, the turbulent inflow condition should be corrected to account for the influence of the developing ice layer on the flow field.

3. In this study a suitable way of providing realistic turbulent inflow to the main simulation was found by using the strong recycling method. However, this methods involves running a tandem simulation that has to meet same the spatial and flow requirements of the main simulation in order to be accurate. At low Reynolds number with an efficient GPU-implementation, computation costs are feasible. However, computational costs will skyrocket for higher Reynolds numbers. More computationally scalable methods can be used, like weak recycling methods [66].

Realistic Freezing Modeling

1. Fluid mixtures in Molten Salt Fast Reactor's typically operate at Prandtl numbers between $7.5 \leq Pr \leq 20$ [67], which is a lot larger than the model's current maximum stable Prandtl number's value at ≤ 1 [61]. A way that the Prandtl numbers stability can be improved in LB model is proposed by Du et al [68]. Here a scaling factor is applied to both the Prandtl number definition and the thermal equilibrium function, leading to stable Prandtl number of up to $Pr = 56.2$. To the author's knowledge, such a treatment has not been applied yet.

2. In the experimental study of Collenteur [30] an adiabatic upper wall was used. To make better comparisons with the experimental reference study, it is recommended to implement an adiabatic wall condition into the FMLBM model.

BIBLIOGRAPHY

[1] United Nations Framework Convention on Climate Change. The paris agreement, 2024. URL https://unfccc.int/process-and-meetings/the-paris-agreement. Accessed July 9, 2025.

[2] NASA. Nasa analysis confirms 2023 as warmest year on record, 2024. URL https://www.nasa.gov/news-release/nasa-analysis-confirms-2023-as-warmest-year-on-record/. Accessed: 2025-05-03.

[3] NASA Earth Observatory. 2024 was the warmest year on record, 2025. URL https://earthobservatory.nasa.gov/images/153806/2024-was-the-warmest-year-on-record. Accessed: 2025-05-03.

[4] World Resources Institute. World greenhouse gas emissions by sector 2021 (sunburst chart), 2021. URL https://climatewatchdata.org/key-visualizations?visualization=4. Accessed: 2025-05-03.

[5] U.S. Energy Information Administration. International energy outlook 2023: Narrative, 2023. URL https://www.eia.gov/outlooks/ieo/narrative/index.php. Accessed: 2025-05-03.

[6] Generation IV International Forum. Welcome to the generation iv international forum, 2025. URL https://www.gen-4.org. Accessed: 2025-05-04.

[7] International Atomic Energy Agency. Nuclear Power Reactors in the World: 2024 Edition, 2024. URL https://www.iaea.org/publications/15488/nuclear-power-reactors-in-the-world-2024-edition. Accessed 13 May 2025.

[8] J. Kenneth Shultis and Richard E. Faw. *Fundamentals of Nuclear Science and Engineering*. Marcel Dekker, Inc., New York, USA, 2002. ISBN 0-8247-0842-3.

[9] SAMOSAFER Project. Concept of the molten salt fast reactor (msfr). https://samosafer.eu/project/concept/, 2025. Accessed: 28 March 2025.

[10] Jan L. Kloosterman. Safety assessment of the molten salt fast reactor (samofar). In Thomas James Dolan, editor, *Molten Salt Reactors and Thorium Energy*, pages 565–570. Woodhead Publishing, 2017. doi: 10.1016/B978-0-08-101126-3.00020-8. URL https://doi.org/10.1016/B978-0-08-101126-3.00020-8.

[11] N. Le Brun, G. F. Hewitt, and C. N. Markides. Transient freezing of molten salts in pipe-flow systems: Application to the direct reactor auxiliary cooling system (dracs). *Applied Energy*, 186(Part 1):56–67, 2017. doi: 10.1016/j.apenergy.2016.09.099. URL https://doi.org/10.1016/j.apenergy.2016.09.099.

[12] Lelio Luzzi, Antonio Cammi, Valentino Di Marcello, and Carlo Fiorina. An approach for the modelling and the analysis of the msr thermo-hydrodynamic behaviour. *Chemical Engineering Science*, 65(16):4873–4883, 2010. doi: 10.1016/j.ces.2010.05.040. URL https://doi.org/10.1016/j.ces.2010.05.040.

[13] Parviz Moin and Krishnan Mahesh. Direct numerical simulation: A tool in turbulence research. *Annual Review of Fluid Mechanics*, 30:539–578, 1998. doi: 10.1146/annurev.fluid.30.1.539.

[14] Cyrus K. Aidun and Jonathan R. Clausen. Lattice-boltzmann method for complex flows. *Annual Review of Fluid Mechanics*, 42:439–472, 2010. doi: 10.1146/annurev-fluid-121108-145519. URL https://doi.org/10.1146/annurev-fluid-121108-145519.

[15] P. L. Bhatnagar, E. P. Gross, and M. Krook. A model for collision processes in gases. i. small amplitude processes in charged and neutral one-component systems. *Physical Review*, 94(3):511–525, 1954. doi: 10.1103/PhysRev.94.511.

[16] Y. H. Qian, D. D'Humières, and P. Lallemand. Lattice bgk models for navier-stokes equation. *Europhysics Letters*, 17(6):479–484, 1992. doi: 10.1209/0295-5075/17/6/001.

[17] Yongguang Cheng and Hui Zhang. A viscosity counteracting approach in the lattice boltzmann bgk model for low viscosity flow: Preliminary verification. *Computers & Mathematics with Applications*, 61(12):3690–3702, 2011. doi: 10.1016/j.camwa.2010.08.078.

[18] D. D'Humières. Generalized lattice boltzmann equations. *Rarefied Gas Dynamics: Theory and Simulations*, 159:450–458, 1992.

[19] Congshan Zhuo, Chengwen Zhong, and Jun Cao. Filter-matrix lattice boltzmann model for incompressible thermal flows. *Physical Review E*, 85(4):046703, 2012. doi: 10.1103/PhysRevE.85.046703. Received 16 September 2011; published 11 April 2012.

[20] Li-Shi Luo, Wenhui Liao, Xuechen Chen, Yanhua Peng, and Weizhong Zhang. Numerics of the lattice boltzmann method: Effects of collision models on the lattice boltzmann simulations. *Physical Review E*, 83(5):056710, 2011. doi: 10.1103/PhysRevE.83.056710.

[21] J. A. Somers. Direct simulation of fluid flow with cellular automata and the lattice-boltzmann equation. *Applied Scientific Research*, 51(1):127–133, 1993. ISSN 1573-1987. doi: 10.1007/BF01082526.

[22] M. Rohde, D. Kandhai, J. J. Derksen, and H. E. A. van den Akker. A generic, mass conservative local grid refinement technique for lattice-boltzmann schemes. *International Journal for Numerical Methods in Fluids*, 51:439–468, 2006. doi: 10.1002/fld.1140. Published online 20 December 2005.

[23] Congshan Zhuo, Chengwen Zhong, and Jun Cao. Filter-matrix lattice boltzmann model for incompressible thermal flows. *Physical Review E*, 85(4):046703, 2012. doi: 10.1103/PhysRevE.85.046703.

[24] Daniel Van Bemmelen. Influence of turbulence on the internal conductivity and total electrical resistance of a carbon black suspension inside a semi-solid flow battery. Ma thesis, Delft University of Technology, 2023.

[25] Pieter van der Spek. Gpu-accelerated large eddy simulation of non-eutectic msfr salt freezing in turbulent channel flow, February 2024. Master's Thesis.

[26] Celeke Bus. Simulating the transient freezing in cooled non-eutectic molten salt channel flow, 2022. Master's Thesis.

[27] Rongzong Huang, Huiying Wu, and Ping Cheng. A new lattice boltzmann model for solid–liquid phase change. *International Journal of Heat and Mass Transfer*, 59:295–301, April 2013. doi: 10.1016/j.ijheatmasstransfer.2012.12.040. URL https://doi.org/10.1016/j.ijheatmasstransfer.2012.12.040.

[28] D. R. Noble and J. R. Torczynski. A lattice-boltzmann method for partially saturated computational cells. *International Journal of Modern Physics C*, 9(08):1189–1201, 1998. doi: 10.1142/S0129183198001026.

[29] Yong Mann Chung and Hyung Jin Sung. Comparative study of inflow conditions for spatially evolving simulation. *AIAA Journal*, 35(2):269–274.

[30] Floor Collenteur. Experimental investigation of solidification phenomena in turbulent and laminar channel flows. Applied physics master thesis, Delft University of Technology, Delft, The Netherlands, February 2024. Supervisors: Dr. Ir. Martin Rohde, Prof. Dr. Ir. Jan Leen Kloosterman, Dr. Stephen de Roode. Daily Supervisor: Ir. Bouke Kaaks.

[31] F.T.M. Nieuwstadt, B.J. Boersma, and J. Westerweel. *Turbulence: Introduction to Theory and Applications of Turbulent Flows*. Springer International Publishing, 2016.

[32] Harrie van den Akker and Rob Mudde. *Fysische Transportverschijnselen: denken in balansen*. TU Delft Open Publishing, 5th edition, 2023. ISBN 978-94-6366-680-0. doi: 10.5074/t.2023.002. URL https://doi.org/10.5074/t.2023.002. Licensed under a Creative Commons Attribution 4.0 International license.

[33] Yongming Zhang. Critical transition reynolds number for plane channel flow. *Applied Mathematics and Mechanics*, 38(10):1401–1410, 2017. doi: 10.1007/s10483-017-2245-6.

[34] V. R. Voller, M. Cross, and N. C. Markatos. An enthalpy method for convection/diffusion phase change. *International Journal for Numerical Methods in Engineering*, 24(1):1–287, 1987. doi: 10.1002/nme.1620240119. URL https://doi.org/10.1002/nme.1620240119.

[35] Timm Krüger, Halim Kusumaatmaja, Alexandr Kuzmin, Orest Shardt, Gonçalo Silva, and Erlend Magnus Viggen. *The Lattice Boltzmann Method: Principles and Practice*. Springer International Publishing, 2017. doi: 10.1007/978-3-319-44649-3.

[36] Vijay Kotra, Sathish Kumar Konidala, N. Anusha, and R. Nageswara Rao. Recent advances and applications of turbulent flow chromatography. *Asian Journal of Chemistry*, 29(4):771–778, 2017. doi: 10.14233/ajchem.2017.20258.

[37] David C. Wilcox. *Turbulence Modeling for CFD, Volume 1*. DCW Industries, La Cañada, California, 3rd, illustrated edition, 2006. ISBN 9781928729082.

[38] Sal Rodriguez. *Applied Computational Fluid Dynamics and Turbulence Modeling: Practical Tools, Tips and Techniques*. Springer Nature Switzerland AG, Cham, Switzerland, 2019. ISBN 978-3-030-28690-3. doi: 10.1007/978-3-030-28691-0.

[39] Philipp Neumann, Hans-Joachim Bungartz, Miriam Mehl, Tobias Neckel, and Tobias Weinzierl. A coupled approach for fluid dynamic problems using the pde framework peano. *Communications in Computational Physics*, 12(1):65–84, July 2012. doi: 10.4208/cicp.210910.200611a.

[40] Sydney Chapman, Thomas George Cowling, and D. Burnett. *The Mathematical Theory of Nonuniform Gases: An Account of Kinetic Theory of Viscosity, Thermal Conduction, and Diffusion*. Cambridge University Press, 1952.

[41] Uriel Frisch et al. Lattice gas hydrodynamics in two and three dimensions. Technical report, Los Alamos National Laboratory (LANL), 1986.

[42] Congshan Zhuo and Chengwen Zhong. Les-based filter-matrix lattice boltzmann model for simulating fully developed turbulent channel flow. *International Journal of Computational Fluid Dynamics*, 30(7-10): 543–553, Nov 2016. ISSN 1061-8562. doi: 10.1080/10618562.2016.1254777. URL https://doi.org/10.1080/10618562.2016.1254777.

[43] K. Suga, Y. Kuwata, K. Takashima, and R. Chikasue. A d3q27 multiple-relaxation-time lattice boltzmann method for turbulent flows. *Computers Mathematics with Applications*, 69(6):518–529, March 2015. doi: 10.1016/j.camwa.2015.01.002.

[44] Mees Wortelboer. Investigating gpu-accelerated double distribution function lattice boltzmann schemes for heat transfer and phase change in turbulent flows. Ma thesis, Delft University of Technology, 2023.

[45] T. Zhang et al. General bounce-back scheme for concentration boundary condition in the lattice boltzmann method. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 85(1):1–14, 2012. ISSN 1539-3755. doi: 10.1103/PhysRevE.85.016701.

[46] Nitin S. Dhamankar, Gregory A. Blaisdell, and Anastasios S. Lyrintzis. Overview of turbulent inflow boundary conditions for large-eddy simulations. *AIAA Journal*, 56(4):1417–1442.

[47] A. Sohankar, C. Norberg, and L. Davidson. Low-reynolds-number flow around a square cylinder at incidence: Study of blockage, onset of vortex shedding and outlet boundary condition. *International Journal for Numerical Methods in Fluids*, 26(1):39–56, 1998. ISSN 0271-2091.

[48] The Julia Programming Language. Julia Documentation. https://docs.julialang.org/, 2025. Accessed: July 8, 2025.

[49] Tom Entes. Development and integration of a gpu-accelerated lattice boltzmann simulation tool for thermal hydraulics with neutronics for multiphysics simulations in molten salt fast reactor cores. Master's thesis, Delft University of Technology, Delft, The Netherlands, February 2025. Supervisors: Dr. Ir. M. Rohde, Dr. Ir. D. Lathouwers, Prof. Dr. Ir. B.J.H. van der Wiel, and Prof. Dr. Ir. C. Vuik.

[50] Robert D. Moser, John Kim, and Nagi N. Mansour. Direct numerical simulation of turbulent channel flow up to $re_\tau = 590$. *Physics of Fluids*, 11(4):943–945, 1999. doi: 10.1063/1.869966.

[51] John Kim, Parviz Moin, and Robert Moser. Turbulence statistics in fully developed channel flow at low reynolds number. *Journal of Fluid Mechanics*, 177:133–166, 1987. doi: 10.1017/S0022112087000892.

[52] Oscar Flores and Javier Jiménez. Hierarchy of minimal flow units in the logarithmic layer. *Physics of Fluids*, 22(7):071704, 2010. doi: 10.1063/1.3464157.

[53] Ricardo Vinuesa, Cezary Prus, Philipp Schlatter, and Hassan M. Nagib. Convergence of numerical simulations of turbulent wall-bounded flows and mean cross-flow structure of rectangular ducts. *Meccanica*, 51(12):3025–3042, 2016. doi: 10.1007/s11012-016-0558-0. URL https://doi.org/10.1007/s11012-016-0558-0. 50th Anniversary of Meccanica.

[54] G. Amati, S. Succi, and R. Piva. Preliminary analysis of the scaling exponents in channel flow turbulence. *Fluid Dynamics Research*, 24(4):201–209, 1999. doi: 10.1016/S0169-5983(99)00022-0.

[55] P. Lammers, K.N. Beronov, R. Volkert, G. Brenner, and F. Durst. Lattice bgk direct numerical simulation of fully developed turbulence in incompressible plane channel flow. *Computers Fluids*, 35(10): 1137–1153, 2006. ISSN 0045-7930. doi: https://doi.org/10.1016/j.compfluid.2005.10.002. URL https://www.sciencedirect.com/science/article/pii/S0045793005001891.

[56] T. Cziesla, H. Braun, G. Biswas, and N. K. Mitra. Large eddy simulation in a channel with exit boundary conditions. NASA Contractor Report 198304 / ICASE Report No. 96-18 NAS1-19480, NASA Langley Research Center, Hampton, Virginia, USA, March 1996. Operated by Universities Space Research Association.

[57] Ao Xu and Bo-Tao Li. Multi-gpu thermal lattice boltzmann simulations using openacc and mpi. *International Journal of Heat and Mass Transfer*, 201:123649, 2023. doi: 10.1016/j.ijheatmasstransfer.2022.123649.

[58] TU Delft. Performance characteristics of the DelftBlue "gpu" nodes. https://doc.dhpc.tudelft.nl/delftblue/perf-V100S/, September 2024. Accessed 9 Jul. 2025.

[59] Nicolas Delbosc, Jean-François Dupuis, Patrick Lallemand, and Li-Shi Luo. Optimized implementation of the lattice boltzmann method on a graphics processing unit towards real-time fluid simulation. *Computers & Mathematics with Applications*, 67(2):462–475, 2014.

[60] Jonas Tölke. Implementation of a lattice boltzmann kernel using the compute unified device architecture developed by nvidia. *Computing and Visualization in Science*, 13(1):29, 2010.

[61] G. Gruszczyński and Ł. Łaniewski Wołłk. A comparative study of 3d cumulant and central moments lattice boltzmann schemes with interpolated boundary conditions for the simulation of thermal flows in high prandtl number regime. *International Journal of Heat and Mass Transfer*, 197:123259, November 2022. ISSN 0017-9310. doi: 10.1016/j.ijheatmasstransfer.2022.123259. URL http://dx.doi.org/10.1016/j.ijheatmasstransfer.2022.123259.

[62] F. Ren, B. Song, and H. Hu. Lattice boltzmann simulations of turbulent channel flow and heat transport by incorporating the vreman model. *Applied Thermal Engineering*, 129:463–471, 2018. doi: 10.1016/j.applthermaleng.2017.10.059.

[63] H. Kawamura, H. Abe, and Y. Matsuo. Dns of turbulent heat transfer in channel flow with respect to reynolds and prandtl number effects. *International Journal of Heat and Fluid Flow*, 21(5):485–491, 2000. doi: 10.1016/S0142-727X(00)00032-5.

[64] Hong Wu, Jiao Wang, and Zhi Tao. Passive heat transfer in a turbulent channel flow simulation using large eddy simulation based on the lattice boltzmann method framework. *International Journal of Heat and Fluid Flow*, 32(6):1111–1119, 2011. doi: 10.1016/j.ijheatfluidflow.2011.07.009.

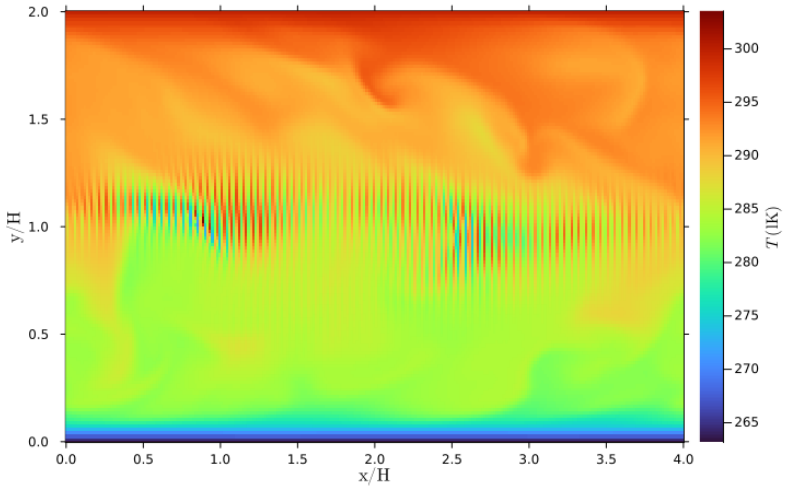
[65] F. Lluesma-Rodríguez, S. Hoyas, and M. J. Perez-Quiles. Influence of the computational domain on dns of turbulent heat transfer up to $re_\tau = 2000$ for $pr = 0.71$. *International Journal of Heat and Mass Transfer*, 122:983–992, 2018. doi: 10.1016/j.ijheatmasstransfer.2018.02.047.

[66] Thomas S. Lund, Xiaohua Wu, and Kyle D. Squires. Generation of turbulent inflow data for spatially-developing boundary layer simulations. *Journal of Computational Physics*, 140:233–258, 1998. doi: 10.1006/jcph.1998.5882.

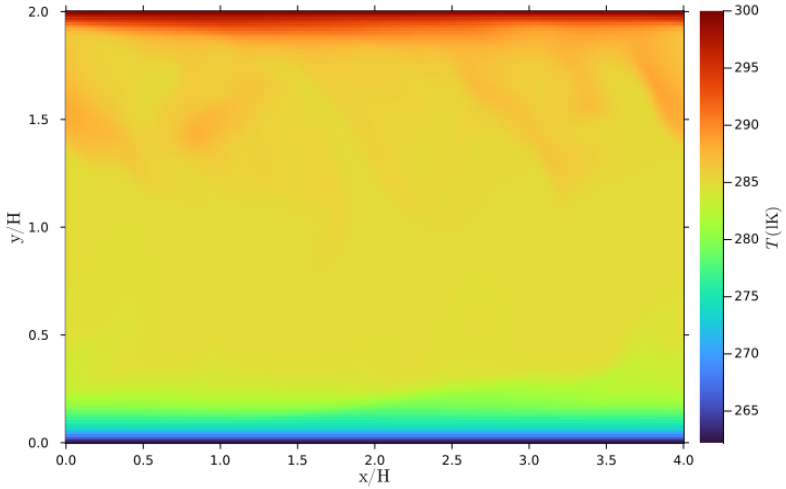
[67] Carlo Fiorina and et al. Thermal-hydraulics of internally heated molten salts and application to the molten salt fast reactor. In *Journal of Physics: Conference Series*, volume 501, page 012030. IOP Publishing, 2014. doi: 10.1088/1742-6596/501/1/012030.

[68] Wenhui Du, Sheng Chen, and Guoqiang Wu. A new lattice boltzmann method for melting processes of high prandtl number phase change materials. *Journal of Energy Storage*, 41:103006, 2021. doi: 10.1016/j.est.2021.103006.

A APPENDIX - THERMAL INSTABILITIES FOR HIGH INITIAL TEMPERATURE GRADIENT



(a) Instable thermal field with $\Delta^+ = 2.8$



(b) Stable thermal field with $\Delta^+ = 2.4$

Figure A.1: Graphs showing two-dimensional temperature snapshots taken at the same number of timesteps.Figure (a) a temperature snapshot displaying thermal instabilities on a grid with $\Delta^+ = 2.8$,(b) a thermal field with gridsize $\Delta^+ = 2.4$ showing no instabilities.